5-2014

Toward Effective Access Control Using Attributes and Pseudoroles

Suhair Alshehri

Toward Effective Access Control Using Attributes and Pseudoroles

by

Suhair Alshehri

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy in Computing and Information Sciences

May 2014

Supervised by

Professor Rajendra K. Raj

The Ph.D. Program in Computing & Information Sciences
B. Thomas Golisano College of Computing & Information Sciences
Rochester Institute of Technology

Certificate of Approval

This is to certify that the doctoral dissertation, "Toward Effective Access Control Using Attributes and Pseudoroles," by Suhair Alshehri has been examined and approved by the dissertation committee as complete and satisfactory for the dissertation requirement for the degree of Doctor of Philosophy.

Rajendra K. Raj, Department of Computer Science, *Dissertation Advisor*

Stanisław P. Radziszowski, Department of Computer Science, *Committee Member*

Carol Romanowski, Department of Computer Science, *Committee Member*

Sumita Mishra, Department of Computing Security, *Committee Member*

Xumin Liu, Department of Computer Science, *Committee Member*

Minseok Kwon, Department of Computer Science, *Committee Member*

Jeff Pelz, Carlson Center for Imaging Science, *Dissertation Defense Chair*

Abstract

Toward Effective Access Control Using Attributes and Pseudoroles

Suhair Alshehri

Supervising Professor: Rajendra K. Raj

Sharing of information is fundamental to modern computing environments across many application domains. Such information sharing, however, raises security and privacy concerns that require effective access control to prevent unauthorized access and ensure compliance with various laws and regulations. Current approaches such as Role-Based Access Control (RBAC), and Attribute-Based Access Control (ABAC) and their variants are inadequate. Although it provides simple administration of access control and user revocation and permission review, RBAC demands complex initial role engineering and makes access control static. ABAC, on the other hand, simplifies initial security setup and enables flexible access control, but increases the complexity of managing privileges, user revocation and user permissions review. These limitations of RBAC and ABAC have thus motivated research into the development of newer models that use attributes and policies while preserving RBAC's advantages.

This dissertation explores the role of *attributes*—characteristics of entities in the system—in achieving effective access control. The first contribution of this dissertation is the design and development of a secure access system using Ciphertext-Policy Attribute-Based Encryption (CP-ABE). The second contribution is the design and validation of a two-step access control approach, the BiLayer Access Control (BLAC) model. The first layer in BLAC checks whether subjects making access

requests have the right BLAC *pseudoroles*—a pseudorole is a predefined subset of a subject's static attributes. If requesting subjects hold the right pseudoroles, the second layer checks rule(s) within associated BLAC policies for further constraints on access. BLAC thus makes use of attributes effectively while preserving RBAC's advantages. The dissertation's third contribution is the design and definition of an evaluation framework for time complexity analysis, and uses this framework to compare BLAC model with RBAC and ABAC. The fourth contribution is the design and construction of a generic access control threat model, and applying it to assess the effectiveness of BLAC, RBAC and ABAC in mitigating insider threats.

To my husband, Ali,
and my daughters, Jonah and Janna.

Acknowledgments

I am endlessly grateful to my advisor, Dr. Rajendra Raj for the guidance, encouragement, and wisdom he has provided through out my PhD program. His guidance has led me in my research, his encouragement has helped me in my academic life, and his wisdom has eased many of the challenges I encountered.

I am also thankful for the invaluable support and help of my committee members: Dr. Stanisław P. Radziszowski, Dr. Carol Romanowski, Dr. Sumita Mishra, Dr. Xumin Liu, and Dr. Minseok Kwon. I am also grateful for the learning experience I received especially from Dr. Radziszowski, Dr. Romanowski and Dr. Mishra. I would like to thank Dr. Jeff Pelz for chairing my defense and reviewing this dissertation.

I am also grateful to my parents, Dhafer and Aysha, sisters and brothers whose love and supplications have always illuminated my way; to my husband, Ali, whose support has given me the strength and courage throughout my studies; to my daughters, Jonah and Janna, whose giggles have eased all the ails; and to my friends whose cheers have brought joys to my life.

Contents

List of Figures

List of Tables

List of Symbols

aR ..... active roles

EAT ..... environment attributes

ob ..... object

OB ..... object dataset

ObAT ..... object attributes

OBR ..... object-to-policy assignment

op ..... operation

OP ..... operation dataset

OpAT ..... operation attributes

p ..... policy

P ..... policy dataset

PA ..... permission-to-object assignment

perm ..... permission

PERM ..... permission dataset

pr . . . . . pseudorole

PR . . . . . pseudorole dataset

PRF . . . . . PseudoRole function

RF . . . . . rule function

r . . . . . role

R . . . . . role dataset

Ru . . . . . access rule

s . . . . . subject

S . . . . . subject dataset

SA . . . . . subject-to-role assignment

SAT . . . . . subject attributes

SPR . . . . . subject-to-pseudorole assignment

List of Abbreviations

AA ..... Attribute Authority

ABAC ..... Attribute-Based Access Control

ADE ..... Access Decision Engine

AES ..... Advanced Encryption Standard

BLAC ..... BiLayer Access Control

CBAC ..... Context-Based Access Control

CP-ABE .... Ciphertext-Policy Attribute-Based Encryption

DAC ..... Discretionary Access Control

ECC ..... Elliptic Curve Cryptography

EHR ..... Electronic Health Record

EMR ..... Electronic Medical Record

FIBE ..... Fuzzy Identity-Based Encryption

GLBA ..... Gramm-Leach-Bliley Act

HIPAA ..... Health Insurance Portability and Accountability Act

IBE ..... Identity-Based Encryption

LBAC ..... Location-Based Access Control

MAC ..... Mandatory Access Control

NIST ..... National Institute of Standards and Technology

PCP ..... Primary Care Physician

PFP ..... Permission Filtering Policy

PHR ..... Personal Health Record

PKC ..... public-key Cryptography

PKE ..... Public-Key Encryption

PKI ..... Public-Key Infrastructure

P-RBAC .... Privacy-Aware Role-Based Access Control

RBAC ..... Role-Based Access Control

RSA ..... Rivest, Shamir and Adleman

SKE ..... Symmetric-Key Encryption

T-RBAC .... Task-Role-Based Access Control

TRBAC ..... Temporal Role-Based Access Control

Chapter 1

Introduction

Information sharing is a basic requirement of modern computing environments in many application domains ranging from healthcare to financial services. Although sharing of information provides several advantages, it raises privacy concerns, particularly when personally identifiable information is involved. That is, such information could potentially identify real-life individuals [55].

Many privacy acts and regulations, including the Health Insurance Portability and Accountability Act (HIPAA), and the Gramm-Leach-Bliley Act (GLBA), impose stringent requirements on handling the protection of such identifiable information. Authentication, access control, and auditing are among the key requirements in such acts and regulations [91]. Although all of these features are fundamental and vital to information and system security, this dissertation focuses on effective access control models.

An *access control* model is typically designed to protect data from (1) unauthorized or improper use and disclosure (*confidentiality*), and (2) unauthorized or improper modification and destruction (*integrity*). Such protection can be achieved by ensuring that decisions for access requests by subjects (users) for protected objects (data) to perform certain operations are regulated by a set of access control

policies.

An *access control* model is conceptually composed of two main components: (1) an enforcement mechanism (or an implementation of a reference monitor), and (2) a decision function. The enforcement mechanism intercepts and examines each access request. It consults with the decision function whether the access request complies with the security policies or not to permit or deny it. The process of making access-control decisions may involve the *identity* of the data owners, *role* played by subjects in an organization or a set of identifiable *attributes* associated with each subject. A generic access control model is depicted in Figure 1.1.

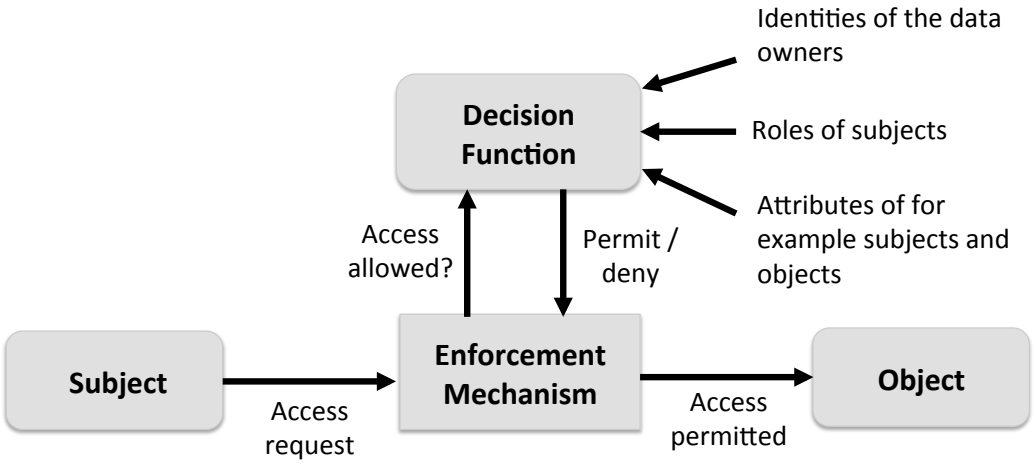

Figure 1.1: A generic access control model, showing the two main components: an enforcement mechanism and a decision function.

1.1 Problem Description

Approaches to access control including Discretionary Access Control model (DAC) [75], Mandatory Access Control model (MAC) [62], Role-Based Access Control model (RBAC) [73] and Attribute-Based Access Control model (ABAC) [84] are inadequate, as explained in the following paragraphs. In DAC, access decisions

are set according to the discretion of owners based on the identity of subjects and objects. On the other hand, in MAC, a system controls access to objects and owners cannot override the controls based on security labels attached to subjects and objects. A label on an object is called a security classification, while a label on a user is called a security clearance. DAC and MAC have several limitations including the lack of scalability and dynamic adaptability to the changes of security policies [12].

In RBAC, access to objects is based on subjects' roles or their job functions. In the education setting, for example, a subject may hold the role "Instructional Faculty." RBAC is widely deployed for its simplicity of administering permissions for large number of users. On the other hand, it is known for its difficulty of defining and structuring roles and inflexibility in dynamically changing environments as it supports coarse-grained and predetermined access control due to the use of roles and the assignments of roles to subjects. RBAC is defined and described in further detail in Chapter 2.

The use of attributes—characteristics of entities to describe and identify them from all others such as name, date of birth, and job title—helps to support finer granularity in access control. Thus, instead of defining access privileges in terms of externally-defined roles, attributes of subjects can be used to specify targeted access rules. The concept of attributes in access control models has been extended to describe objects and environment conditions in addition to subjects' properties.

Here is an example from the education domain of three access rules with three different levels of granularity to demonstrate the power of the fine-granularity provided by the use of attributes in defining access-control rules:

1. **Coarse:** Faculty members are allowed to open the department office door.

2. **Fine:** Faculty members in the Computer Science department are allowed to open the department office door.

3. **Finer:** Faculty members in the Computer Science department are allowed to open the department office door during office hours if they are assigned to that office.

Thus, as more attributes are included in access-control rules, the more specified and targeted the access rules are going to be. In short, attributes allow for extremely fine-grained access control.

ABAC, which is based on attributes, is a relatively recent yet promising access control model [27, 84], which employs attributes and access policies to define privileges of subjects. In ABAC, access requests compared against attributes within these policies to determine whether to allow or deny access requests. ABAC overcomes the RBAC problem of defining and structuring roles as it (ABAC) uses attributes for making access-control decisions. Because ABAC uses attributes, it supports fine-grained access control policies that can be evaluated dynamically in real-time, and are flexible enough to support environments with frequently changing user permissions.

Here is another example from the education domain to demonstrate the flexibility provided by the use of attributes in access control. A faculty member works in a college and he is defined with a set of attributes. One of these attributes is *<Job Title, Instructional Faculty>*. His access privileges in ABAC stem from such attributes, not from the notion of a single role (as in RBAC). So assuming this faculty member gets promoted to department chair, the faculty member's access privileges will need to be updated accordingly.

In ABAC, this change requires only a simple modification to the value of the attribute job title from *<Job Title, Instructional Faculty>* to *<Job Title, Department Chair>*. The faculty is immediately able to access all the information needed to perform the new chair duties. No adjustments to existing access policies or rules are needed, and no modifications to subject membership into roles as in RBAC is

needed.

ABAC, however, is more cumbersome in terms of the number of rules that need to be evaluated for access-control decisions. For n attributes, ABAC may require up to 2^n possible rules. Also, management of privileges and permission review for a particular user are difficult to perform as a large set of rules must be executed [37].

Table 1.1 describes how access-control decisions are evaluated in the access control models discussed previously. In MAC and DAC, for example, access-control decisions are respectively based on security labels of subjects and objects, and identities of subjects. In RBAC, access-control decisions are made based on the mapping between subjects and roles, and roles and permissions. In ABAC, access-control decisions are evaluated using attributes.

Table 1.1: Decision Evaluation in access control models. Access-control decisions in MAC, DAC, RBAC and ABAC are made respectively according to security labels of subjects and objects, identities of subjects, assignments of subjects to roles and roles to permissions, and attributes.

Model	U	SL	R	A	**Permission Mapping**
MAC	0	1	0	0	$SL \rightarrow$ Permission
DAC	1	0	0	0	$U \rightarrow$ Permission
RBAC	1	0	1	0	$U \rightarrow R \rightarrow$ Permission
ABAC	0	0	0	1	$A_1,\ldots,A_n \rightarrow$ Permission
U=User; SL=Security Label; R=Role; A=Attribute					

Due to the limitations of both RBAC and ABAC, research therefore is needed to understand how attributes could be used effectively in designing newer access control models to improve the process of making access-control decisions. This dissertation first examines the use of attributes and policies in Ciphertext-Policy

Attribute-Based Encryption (CP-ABE) [13]. In CP-ABE, subjects' private keys are labeled with sets of attributes and objects are encrypted and associated with access structures or rules consisting of AND and OR gates. A subject's private key can decrypt a particular encrypted object or ciphertext only if the attribute set of the subject's key satisfies the access rule associated with the ciphertext. Details about this approach and its viability for effective access control are presented in Chapter 3.

The National Institute of Standards and Technology (NIST) [37] recently called for the development of a policy-enhanced RBAC model that includes the use of attributes while maintaining the advantages of RBAC. Three possible mechanisms were identified: (1) *dynamic roles*, in which attributes are used to dynamically assign roles to subjects, (2) *attribute centric*, in which roles are defined as another attribute of subjects, and (3) *role centric*, in which attributes are used to constrain the permissions assigned to roles.

These suggested approaches also suffer drawbacks. The first approach inherits all limitations of RBAC including the lack of granularity and dynamic adaptability. The second approach discards all the advantages of RBAC and inherits all the disadvantages of ABAC. The third approach lacks the flexibility of ABAC because the maximum permission sets are constrained by the use of roles.

Given the likely impact of a solution to this call by NIST, the dissertation examines how attributes could be used effectively in designing newer access control models while preserving RBACs advantages. Toward achieving this goal, the dissertation proposes a new variant to RBAC and ABAC, the BiLayer Access Control (BLAC) model [7]. The design of the BLAC model is inspired by several aspects of CP-ABE, including the association of private keys and attributes and ciphertexts and access structures. The complete definition and description of the BLAC model is provided in Chapter 4.

1.2 Contributions

The major objective of this dissertation has been the exploration of attributes and policies for the design of effective access control models. As a consequence, the contributions of this dissertation span several distinct yet related issues around access control space as listed below.

- This dissertation shows how attributes and policies in CP-ABE [13] can address observed inadequacies of current approaches to access control and standard encryption techniques, through the design and development of a cloud-based system for electronic health records (EHRs) as described in Alshehri [6]. Specifically, the dissertation shows that CP-ABE provides promising performance to enforce access control in information sharing environments.

- The dissertation addresses the problem of improving access control via the use of attributes for making access-control decisions while retaining the advantages of RBAC. As a result, the dissertation proposes the design and validation of the BLAC model, which is described in Alshehri [7]. Unlike current access control approaches, the BLAC model uses the concept of *pseudoroles* and performs access control in a two-step evaluation procedure. A *pseudorole* is informally defined as a set of values of static attributes of subjects. That is, a pseudorole is not a real role as traditionally defined in RBAC. The first layer in the two-step evaluation process checks access requests to verify whether requesting subjects have the *right* pseudoroles specified in the BLAC policies of the requested objects. If requesting subjects hold the right pseudoroles, the second layer checks rule(s) within the associated BLAC policies for additional constraints on access.

- This dissertation designs and defines an evaluation framework that theoretically and quantitatively analyzes and compares the effectiveness of access

control models. The framework is composed of five core functions of access control models: authorization decision-making, policy modification, permission modification, revocation and permission review. The functions underlying access control models are theoretically analyzed for their time complexity, and these results are then used to compare several access control models.

- The dissertation examines how shared information can be protected from unauthorized or improper use, disclosure, alteration, and destruction by insiders, as described in Alshehri [5]. The dissertation uses a generic access control system to construct a threat model to assess the effectiveness of current access control mechanisms, including RBAC, ABAC and BLAC, in mitigating insider threats. This assessment allows the contrast of the BLAC model against the two major access control models, RBAC and ABAC.

1.3 Dissertation Organization

The rest of the dissertation is organized as follows. Chapter 2 presents the background in access control models and related work in this space. Chapter 3 presents one approach for using attributes for secure access control via CP-ABE. Chapter 4 presents the BLAC model, which is a major contribution of this dissertation. Chapters 5 and 6 present the evaluation framework and the generic access control threat model respectively, and apply them to assess the effectiveness of the BLAC model. Chapter 7 concludes the dissertation and discusses future research directions.

Chapter 2

Background

This chapter reviews the foundational and related work in the area of access control, and motivates the need for developing an access control model that uses attributes in access decisions while preserving the benefits of RBAC.

2.1 Role-Based Access Control (RBAC)

The term RBAC was first introduced and formally modeled in the early 1990s by Ferraiolo and Kuhn [22]. Subsequently, Sandhu et al. proposed frameworks for four RBAC models to meet various levels of complexity requirements [73]. These models are core RBAC, hierarchical RBAC, static separation of duty relations, and dynamic separation of duty relations. The core RBAC model introduces the basic and minimum components of RBAC in which subjects are assigned to roles and permissions are assigned to roles. The hierarchical RBAC extends the core model by introducing the support for role hierarchies to define an inheritance relation among roles. Separation of duty relations are used to enforce conflict of interest policies to restrict subjects from exceeding a reasonable level of authority to perform a task. This is can be done through the static separation of duty or dynamic

separation of duty relations. The former enforces strong constraints on the assignment of subjects to roles; for example, if one role requests an order and another approves it, the subject should not be assigned both roles. The latter enforces the constraints on the assignment of subjects to roles in addition to the context such as the time. All these models have the same limitations of the core RBAC that Chapter 1 discusses.

RBAC has been widely investigated to meet various applications' requirements. The rest of this section presents the basic RBAC model and discusses the main extensions of RBAC.

2.1.1 The Basic RBAC Model

The NIST model for RBAC [23] was adopted in this dissertation focusing, for the sake of simplicity, on the core RBAC that introduces the required components for any RBAC model [74]. The basic elements of RBAC are illustrated in Figure 2.1.

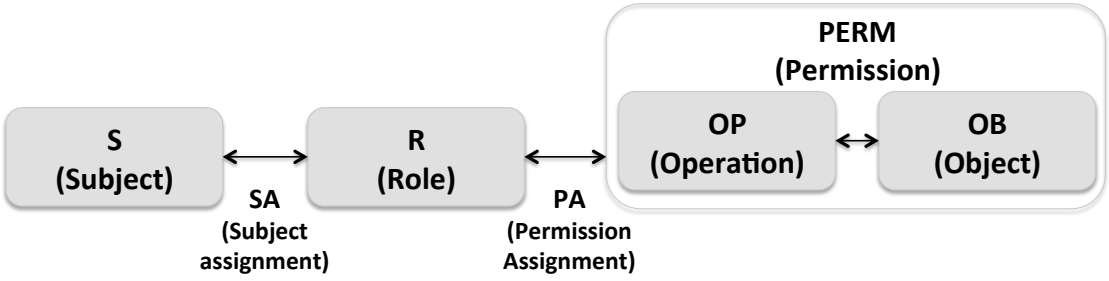

Figure 2.1: The basic components of the core RBAC, containing five data sets: subject, role, object, operation and permission, and two relations: subject assignment and permission assignment.

RBAC is defined in terms of five basic data sets: *S* (subjects), *R* (roles), *OB* (objects), *OP* (operations), and *PERM* (permissions), and two relations: *SA* (subject-to-role assignment) and *PA* (permission-to-role assignment).

Subjects are assigned to roles, and roles are associated with permissions that define which operations can be performed over which objects. Subjects are defined as individuals, and roles represent job functions within the organization. In the context of healthcare, roles could be "doctor," "nurse," and "staff." Permissions could be "read a record" and "write a record."
The following definition of RBAC is adopted from Ferraiolo et al. [23].

Definition 2.1.1. [RBAC]

- *S, R, OP, and OB are subjects, roles, operations, and objects, respectively.*
- $SA \subseteq S \times R$, *a many-to-many mapping subject-to-role assignment relation.*
- *assigned-subjects*$(r) = \{s \in S \mid (s,r) \in SA\}$, *the mapping of role r onto a set of subjects.*
- *PERM* (*the set of permissions*) $\subseteq \{(op,ob) \mid op \in OP \bigwedge ob \in OB\}$.
- $PA \subseteq PERM \times R$, *a many-to-many mapping permission-to-role assignment relation.*
- *assigned-permissions*$(r) = \{perm \in PERM \mid (perm,r) \in PA\}$, *the mapping of role r onto a set of permissions.*

2.1.2 RBAC Extensions

RBAC is a good starting model for researchers to advance access control models due to its widespread adoption and formal definition. These extensions are needed to meet requirements in specific domain areas. One of the significant extensions of RBAC is Context-Based Access Control (CBAC) in which access decisions are based on the context of the request in addition to requesters' roles concerning the temporal constraints, as in Temporal Role-Based Access Control (TRBAC) [11,33] or location constraints, as in Location-Based Access Control (LBAC) [19,68]. Other

major extensions include Task-Role-Based Access Control (T-RBAC) [61, 72], and Privacy-Aware Role-Based Access Control (P-RBAC) [47, 59].

In TRBAC, the use of permissions assigned to roles can be enabled only during specific temporal periods which can be fixed or periodic. Bertino et al. [11] proposed a TRBAC model that supports temporal constraints on role activation and deactivation. Thus, roles can be in two states: active, in which the associated roles can be used, and non-active, in which associated roles cannot be used. The model also defines temporal dependencies among role activation and deactivation which can be expressed through triggers. For example, when a role is activated, a trigger can be set up to activate another role. Joshi et al. [33] extended TRBAC to a generalized TRBAC (GTRBAC) that introduced temporal constraints user-role assignments, and role-permission assignments in addition to roles.

In LBAC, access decisions are based on roles and constraints on users' locations to meet the increasing security requirements by mobile applications and users. Thus, access requests are granted, for example, when users are in a specific location within an organization's premises or a location that satisfies certain security properties. Damiani et al. [19] extended RBAC by supporting the notation of spatial roles for geographically bounded organizational functions. The boundary of a role is defined as a feature such as road, city or hospital. This boundary specifies the spatial extent in which the user has to be located in to use the role. Ray and Toahchoodee [68] extended LBAC to support spatial and temporal constraints on users and objects.

In T-RBAC, access decisions are based on roles and tasks that represent fundamental units of business activity in enterprise environments. A role can be "Sales Manager," and a task can "review sales results." Oh and Park [61] extended RBAC by incorporating the notation of tasks in addition to roles to evaluate access requests. In their model, permissions are assigned to tasks, and tasks are assigned to

roles. Sainan [72] extended the model by Oh and Park by introducing constraints rules between users and roles, and roles and tasks.

In P-RBAC, privacy policies are incorporated in access control systems, thus access to private and sensitive data is considered when making access decisions. Ni et al. [59] extended RBAC to support privacy policies through introducing purposes, conditions and obligations. Purposes are the reasons for data access. Obligations are actions that must be performed after an access has been executed. Conditions are prerequisites needed to be met before any action can be executed. In the proposed model, users are assigned to roles, and permissions are assigned to roles. However, the main difference here is the structure of the privacy permissions.

Masoumzadeh and Joshi [47] extended the model presented by Ni et al. by treating a purpose as an intermediary entity between a role and permission entities. The model also defines constraints and obligations as conditions on assignment of permissions to purposes. In Masoumzadeh and Joshi's model, users are assigned to roles, purposes are assigned to roles, permissions are then assigned to purposes, and conditions are assigned to permission.

2.2 Attribute-Based Access Control (ABAC)

Access decisions in ABAC are based on the use of attributes that are compared against rules created by security administrators. These attributes are associated with entities such as subjects, objects and the context of access requests. To the best of the author's knowledge, the first work that explicitly introduced a model for ABAC was by Yuan and Tong [101]. However, the use of attributes and policies was introduced earlier through the use of security credentials [14, 15, 63].

2.2.1 The Basic ABAC Model

The concept of ABAC is based on the use of attributes (subjects, environments, and objects) and access rules [27,101]. In addition, attributes are sets of key-value pairs to describe subjects, objects, and environments. The basic components of ABAC are shown in Figure 2.2.

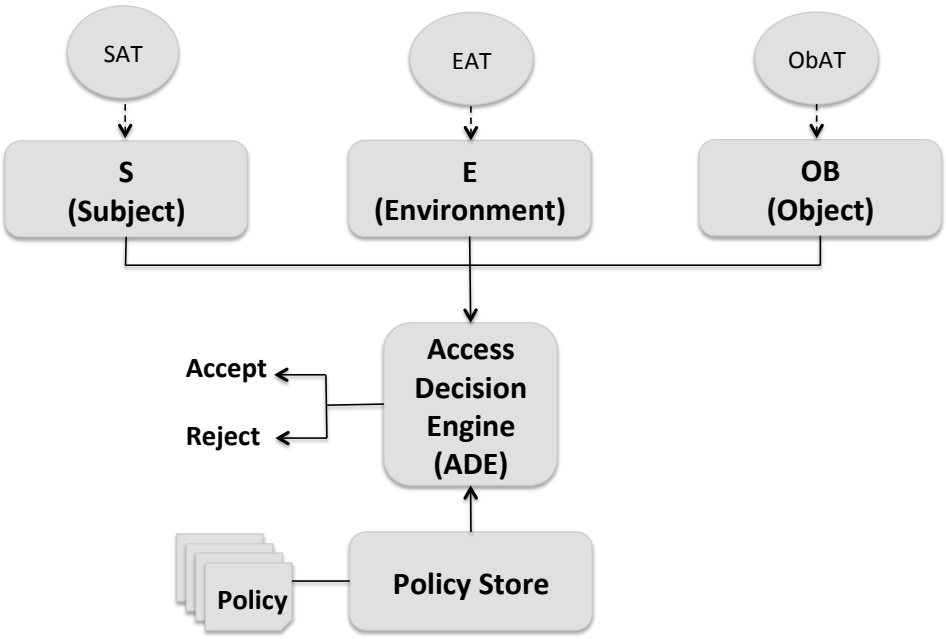

Figure 2.2: The basic components of ABAC, employing attributes of subjects, objects, environment conditions and access policies. SAT, EAT and ObAT represent subject attributes, environment attributes and object attributes respectively.

Here, *<Provider, Doctor>* and *<Department, Cardiology>* could be subject attributes. *<Patient Name, Bob>* and *<Document Type, Summary Of Care>* could be object attributes. *<Access IP=192.123.*.*>*, on the other hand, could be an environment attribute. Here, an example access rule could be *A doctor who works in Cardiology can access Bob's summary of care report from a computer on a specific subnet of the hospital's network.*

Attributes can be static or dynamic, depending on how frequently their values change. Access rules, i.e. policies, are defined using attributes. Requesters must possess attributes that satisfy relevant rules to be granted access to requested objects.

The following definition of ABAC is adopted from Yuan and Tong [101].

Definition 2.2.1. [ABAC]

- *S, OB, and E are subjects, objects, and environments, respectively.*

- *SAT_k $(1 \leq k \leq K)$, $ObAT_m$ $(1 \leq m \leq M)$, and EAT_n $(1 \leq n \leq N)$ are the pre-defined attributes for subjects, objects, and environments, respectively.*

- *$ATTR(s)$, $ATTR(o)$, and $ATTR(e)$ are attribute assignment relations for subject s, object o, and environment e, respectively:*
  $ATTR(s) \subseteq SA_1 \times SA_2 \times \ldots \times SA_K$
  $ATTR(o) \subseteq OA_1 \times OA_2 \times \ldots \times OA_M$
  $ATTR(e) \subseteq EA_1 \times EA_2 \times \ldots \times EA_N$.

- *A policy rule that decides whether subject s can access object o in a particular environment e, is a boolean function of s, o, and e's attributes:*
  Rule: can-access $(s, o, e) \leftarrow f(ATTR(s), ATTR(o), ATTR(e))$
  i.e. given all the attribute assignments of s, o, and e, if the function's evaluation is true, then the access to the object is granted; otherwise the access is denied.

- *A policy store, as shown in Figure 2.2, may consist of a number of policy rules. The policy decision engine evaluates the applicable policy rules in the policy store for making the authorization decision.*

2.2.2 ABAC Extensions

Yuan and Tong [101] discussed the challenges of access control for web services. They proposed an ABAC model based on subject, object, and environment attributes. They also described the model in terms of authorization architecture and policy formulation. The ABAC model here consists of two sub-models: the policy model that defines the policies, and the architecture model that applies the policies to web services. In the policy model, a policy rule is a Boolean function that decides on whether a subject s can access a resource r in a particular environment e. For example,

$$Rule\colon \textit{can-access}(s, r, e) \leftarrow f(ATTR(s), ATTR(r), ATTR(e)).$$

Given the attributes of s, r, and e, if the function's evaluation returns true, then the access to the resource is granted; otherwise the access is denied. The ABAC model described by Yuan and Tong, however, does not consider operations on resources and does not provide any implementation details of the policy evaluation.

Subsequently, many researchers presented other models for ABAC for different applications and domains [4,38,40,79,102]. Hai-bo and Fan described a model for ABAC for web services [79] that evaluate access requests to web services based on the attributes that requesters hold. Hai-bo and Fan did not provide any algorithms or implementation specifications.

Alipour et al. [4] extended Yuan and Tong's ABAC model for SOA environments. Zhu and Smari [102] presented an ABAC model for collaboration environments. Their model incorporated trust and privacy preserving concerns by monitoring subjects' previous behavior and ensuring that the purpose of access corresponds to the intended use of the requested objects.

Lang et al. [38] described an Attribute-Based Multipolicy Access Control Model for Grid Computing that addressed the problem of supporting multiple types of policies without changing the policy description and evaluation mechanism. Also,

Li et al. presented an ABAC model for ubiquitous e-business environments [40].

HP Praesidium Authorization Server was a commercial implementation of a rule-based access control model [34]. It centralizes authorization rules and privileges in a single database shared by multiple applications. Rules and privileges are specified by an organization to define access rules in the former and to specify access limits for subjects in the latter. Thus, when a subject requests to perform a transaction, transaction attributes input from the application and privilege attributes input from the privilege are evaluated against an access rule for each privileges of the requesting subject. If true, access is granted, if false, continue; if all privileges specified for the requesting subject have been evaluated to false, access is denied.

Although all the previously mentioned research work was important to advance ABAC models, it did not consider the complexity of ABAC in terms of privilege management, user permissions, and permission review for a user.

2.3 Revising the NIST-RBAC Model

Due to the limitations of RBAC and ABAC, Kuhn et al. [37] recently announced a NIST initiative that focuses on addressing the need of including attributes in access decisions while preserving the benefits of RBAC. Kuhn et al. suggested combining RBAC and ABAC to leverage the advantages of each. They discussed three possible combination strategies for integrating attributes into RBAC: (1) dynamic roles, (2) attribute centric, and (3) role centric. The rest of this section discusses these approaches and their drawbacks.

2.3.1 Dynamic roles

In this approach, attributes are used to dynamically assign subjects to roles. This dynamic role approach has been studied by Al-Kahtani et al. [2] and Huang et al. [29]. Although these models solved the problem of assigning subjects into roles, they inherit the major limitations of RBAC described earlier in chapter 1, including the lack of granularity and flexibility and dynamic adaptability. Al-Kahtani et al. [2] introduced a rule-based RBAC model that uses rules defined by an organization. These rules are used to automatically assign users to roles taking into account users' attributes and constraints on using roles.

Huang et al. [29] presented an integrated model of RBAC and ABAC. The idea behind the model is the composition of two layers: *aboveground* and *underground*. The former constructs a traditional RBAC model extended with environment constraints. The latter uses attribute-based user-role assignment policies and role-permission assignment policies to automate user-role assignment and role-permission assignment functions for the aboveground level or the traditional RBAC model. Again, neither of the models address the problem of granularity and dynamic adaptability, which is the focus of this work.

2.3.2 Attribute-centric Approach

In the attribute centric approach, roles are defined as attributes of subjects. Thus, roles are not assigned to a set of permissions as in RBAC. This approach discards all the advantages of RBAC and inherits all the disadvantages of ABAC discussed earlier in Section 1. Basic ABAC models presented earlier can be used to implement an attribute-centric ABAC model.

2.3.3 Role-centric Approach

In the role-centric approach, attributes are used to constrain the permissions assigned to roles. Although this approach retains the ability of RBAC to determine the maximum set of permissions for a single user, it lacks ABAC's flexibility because the maximum permission sets are constrained by the use of roles. Jin et al. [31] present a model for a role-centric approach that incorporates RBAC basic elements in addition to user attributes, object attributes, and permission filtering policy (PFP).

The PFP constrains the available set of permissions based on user and object attributes. The *avail-session-perm* function in basic RBAC returns the permission set associated with the roles activated in a given session. However, in this model, the function returns the maximum permission set for a given session. The permission set is then reduced by the PFP. For each of the permissions in the permission set, there is a function that maps objects to subsets of Boolean filter functions, which are based on the use of object attributes. These filter functions are compared against the associated permission within the permission set returned by *avail-session-perm*. If any of these functions returns false, the permission is blocked and removed from the available permission set for the given session.

2.4 Chapter Summary

This chapter presented the background and reviewed the related word in the area of access control to set the stage for this dissertation. It defined and reviewed the basic access control models, RBAC and ABAC and their extensions. The chapter also discussed the call by NIST to advance access control models to use attributes and policies for making access decisions while preserving RBAC's advantages.

The chapter presented the approaches proposed by NIST and discussed their limitations.

Chapter 3

Access Control using Attribute-Based Encryption

Cloud computing has been viewed as an appropriate platform to deploy applications in various domains for its cost-effective services (including data management and storage, and computational resources) and features (portability, reliability, scalability, and elasticity) delivered by cloud service providers [18]. Despite these attractions to cloud-based systems, cloud service providers are not trusted to store private information unencrypted, for example EHRs, even when traditional access control models are in place [18]. This is because unencrypted information stored in the cloud is vulnerable to unauthorized disclosure or alteration. Thus, end-to-end information encryption (including storage and transmission) must be required in cloud-based systems.

Standard encryption techniques, however, are not well suited for EHR and similar systems, especially in cloud-based settings, as multiple subjects have have overlapping but distinct roles and attributes in accessing EHR and similar protected data. The following paragraphs review standard encryption techniques and discuss their unsuitability for such cloud-based systems.

- **Symmetric-Key Encryption (SKE).** These techniques, for example, AES (Advanced Encryption Standard), typically provide secrecy between two parties. SKE uses a shared key k for encryption and decryption;

  $$M = D_k(E_k(M)).$$

  In the above function, E is the encryption function and it uses the shared key k to encrypt the message M. D, on the other hand, is the decryption function and it uses the shared key k to decrypt the encrypted message. For two parties, say Alice and Bob to use SKE, they first need to agree on a shared key k. Alice then encrypts M using the the encryption algorithm E and key k to obtain the ciphertext $C = E_k(M)$. Alice then sends C to Bob. Bob uses the decryption algorithm D and the same key k to recover the message $M = D_k(C)$. SKE techniques are usually efficient but introduce complexity in systems as additional mechanisms are required to apply access control. Also, parties who want to use SKE need first to agree on the shared key via key exchange methods, for example public-key cryptography, such as Diffie-Hellman key exchange. As SKE uses a shared key among parties, if the shared key is compromised, all encrypted information is compromised.

- **Public-Key Encryption (PKE).** Techniques such as RSA (Rivest, Shamir and Adleman) provide methods for shared key distribution, authentication and non-repudiation. In PKE, each subject has a pair of keys: a public key pk that is known to everyone, and a secret key sk that is only known to the subject. PKE uses the public key pk for encryption and the secret key sk for decryption:

  $$M = D_{sk}(E_{pk}(M)).$$

  In the above function, E, the encryption function, uses the public key pk to encrypt the message M. D, the decryption function, uses the secret key sk to decrypt the encrypted message. Again, for Alice and Bob to use PKE, assume

Bob has the pair of keys: pk and sk, and that Alice wants to encrypt a message M for Bob. Alice encrypts M using the encryption algorithm E and Bob's public key pk to obtain the ciphertext $C = E_{pk}(M)$. Alice sends C to Bob who uses the decryption function and his secret key sk to compute $M = D_{sk}(C)$ to recover the message M. PKE are are not practical for secure storage due to the requirement for an expensive public-key infrastructure (PKI) to be maintained for distributing and managing public keys and digital certificates for subjects.

The above inadequacies with standard access control and encryption techniques in supporting cloud-based systems motivate the investigation of other approaches. This work builds on CP-ABE [13] that utilizes attributes for accessing and handling secure data.

This chapter shows how attributes and policies in CP-ABE can be used effectively in designing access control systems, which improve the process of making access-control decisions through the design and development of a secure attribute-based system, as described in Alshehri [6]. Particularly, the dissertation shows that CP-ABE provides promising performance to enforce access control in information-sharing environments.

This chapter first presents the relevant background in cryptography. It then describes the proposed design for the cloud-based system using CP-ABE in the context of healthcare domain. Although the focus is on EHR systems, the proposed design is general and may be applied to other domains where users have multiple overlapping roles and attributes in data access and handling, for example, financial services and critical infrastructure protection systems. This chapter then provides an assessment of performance of secure cloud-based EHR systems to validate the proposed approach. Finally, the chapter discusses the related work in cryptography and EHR systems.

3.1 Cryptography Preliminaries

This section reviews the needed background in cryptography to pave the way to the discussion of the secure attribute-based access system in the next section.

3.1.1 Bilinear Maps

Bilinear Maps, or pairings, construct a relationship between two cryptographic groups leading to new schemes. Boneh and Franklin's pairing-based encryption scheme [16] was the first to construct a fully functional Identity-Based Encryption (IBE) that is based on bilinear mapping on groups of elliptic curves over finite fields. IDE, however, was originally proposed by Adi Shamir in 1984 [77], by using a public key as an arbitrary string to identify a user. Seventeen years later, Boneh and Franklin built an elegant scheme using bilinear maps (Weil pairing) over elliptic curves and finite fields [16]. The abstract definition of bilinear pairing is presented in Definition 3.1.1.

Definition 3.1.1. [Bilinear Maps]
Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be cyclic groups of prime order p; g a generator of $\mathbb{G}_1$. e is a bilinear map, e: $\mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, where $|\mathbb{G}_1| = |\mathbb{G}_2| = p$. The bilinear map e has three properties:

- *Bilinearity:* $\forall\ P,Q \in \mathbb{G}_1, \forall\ a,b \in \mathbb{Z}_p^*, e(aP, bQ) = e(P, Q)^{ab}$,
- *Non-Degeneracy:* $P \neq 0 \Rightarrow e(P, P) \neq 1$,
- *Computability: e is efficiently computable.*

The computable pairing function *e* must be efficient. This means, in practice, when a system using *e* is deployed, *e* must be fast enough so a subject using the system must not notice any slowdown [42].

3.1.2 Elliptic Curve Cryptography

Elliptic Curve Cryptography (ECC) is a type of public-key Cryptography (PKC) that is based on the algebraic structure of elliptic curves over finite fields. The relationship between elliptic curves and cryptography was first introduced by Hendrik Lenstra [39] in 1987 when he presented the elliptic curve factorization method for integer factorization which employs elliptic curves. This discovery inspired researchers to investigate other approaches to incorporate elliptic curves into cryptography. In 1985, Koblitz [35] and Miller [54] independently suggested the use of elliptic curve over a finite field for cryptographic schemes. The security of these cryptographic schemes rely on the hardness of the elliptic curve discrete logarithm problem. The elliptic curves over finite field and the elliptic curve discrete logarithm problem are defined defined in Definition 3.1.2 and Definition 3.1.3 respectively.

Definition 3.1.2. [Elliptic Curves over Finite Field]
Let F_p be a finite field where $p > 3$ is a prime, and a, $b \in F_p$ such that $4a^3 + 27b^2 \neq 0 \ (mod\ p)$. An elliptic curve $E[F_p]$ is the set of solutions, called points $P = (x, y)$ to the equation $y^2 = x^3 + ax + b \ (mod\ p)$, together with the point at infinity 0. Addition of points on ECC is defined to form the so-called elliptic curve group. For $P \in E[F]$, nP denotes $P + P + \ldots + P$, n times.

Definition 3.1.3. [Elliptic Curve Discrete Logarithm Problem]
If E is an elliptic curve over a field F, then the elliptic curve discrete logarithm to base $Q \in E[F]$ is the problem of finding an $n \in Z$ such that $P = nQ$ for a given $P \in E[F]$.

The advantage of using ECC is that it can achieve RSA-equivalent security with

a much smaller group. For example, a 163-bit key in ECC is considered to be as secure as 1024-bit key in RSA. This results in smaller key sizes and faster implementations. These features allow ECC to be suitable to be used on compact platforms such as smart phones and smart cards [36].

The PBC (Pairing-Based Cryptography) library [43] is designed to perform implementations for pairing-based cryptosystems including elliptic curve generation, elliptic curve arithmetic and pairing computation. The PBC library supports seven types of elliptic curves for pairing-based cryptography over finite fields; however, these curves can be grouped into supersingular curves and ordinary curves. These types of curves are as follows [42, 43].

1. **Type A Curves:** Let E be an elliptic curve of the form

   $$y^2 = x^3 + ax \text{ for any } a$$

   over a finite field F_p for some prime $p = 3 \ (mod \ 4)$. 1, -1 and -3 are good choices for a to achieve the best performance. $E[F_p]$ is supersingular, and thus $\#E[F_p] = q + 1$ and $\#E[F_{p^2}] = (q + 1)^2$. Type A curves have an embedding degree of 2.

2. **Type B Curves:** Let E be an elliptic curve of the form

   $$y^2 = x^3 + b \text{ for any } b$$

   over a finite field F_p for some prime $p = 2 \ (mod \ 3)$. Typically, b = 1 or -1. Similar to the type A curves, $E[F_p]$ is supersingular, and thus $\#E[F_p] = q + 1$ and $\#E[F_{p^2}] = (q + 1)^2$. Again, these curves have an embedding degree of 2. Type B curves are unimplemented in the PBC library as they provide no significant advantage over type A curves [46].

3. **Type C Curves:** Let E be an elliptic curve of the form

   $$y^2 = x^3 + 2x + 1 \text{ over } F_{3^l}$$

   and the form

$$y^2 = x^3 + 2x - 1 \text{ also over } F_{3^l}.$$

Type C curves are unimplemented as well as few useful cryptographic curves are found and pairings on these type of curves are vulnerable to certain discrete log attacks [42].

4. **Type D Curves:** These are ordinary curves with embedding degree of 3, 4 or 6, however, 6 is the most useful. Type D curves are constructed using the complex multiplication equation

$$DV^2 = 4p - t^2$$

Supposing D, V, p, t are integers and p is prime and $D > 0$, no square of an odd prime divides D and $D = 0, 3 \ mod \ 4$. Type D curves are constructed by substituting $p = x^2 + 1$, $t = \pm x + 1$ and $U = 3x \pm 1$ into the complex multiplication equation, and solving the resulting equation. The resulting curves are of order $p \mp x$.

5. **Type E Curves:** These are also ordinary curves with embedding degree of 1. Type E curves can be constructed by setting $t = 2$, $D = 7$. Let r be a positive integer and suppose $p = 28r^2h^2 + 1$ is prime for some h. Then the complex multiplication equation becomes

$$7V^2 = 4(28(rh)^2 + 1) - 4$$

After these substitutions, the equation is solved to $V = 4rh$. As $H_7(x) = x + 3375$, if $k = 3375/(17283375)$ in F_p then the produced curve is of the form

$$y^2 = x^3 + 3kx + 2k$$

of order $p - 1$ or $p + 3$. If this curve has the order $p + 3$, then use instead the curve of the form

$$y^2 = x^3 + 3kc^2x + 2kc^3,$$

where c is any quadratic nonresidue $\in F_p$.

6. **Type F Curves:** These are also ordinary curves with embedding degree of 12.

Type F curves can be constructed by setting $p = 36x^4 + 36x^3 + 24x^2 + 6x + 1$, $t = 6x^2+1$, and $D = 3$. Then the complex multiplication equation has solution

$$V = 6x^2 + 4x + 1,$$

and $p+1-t \mid p^{12}-1$. To construct a type F curve, any value of x can be chosen, then p should be checked if it is a prime for x, and that $n = p(x)t(x) + 1$ has a large prime factor r. If these statements are true, then different values of k can be tried until a random point of $y2 = x3 + k$ has order n is found.

7. **Type G Curves:** These are also ordinary curves with embedding degree of 10. Type G curves can be constructed by setting $p = 25x^4 + 25x^3 + 25x^2 + 10x + 3$, $t = 10x^2 + 5x + 3$, and D is squarefree and $D = 43, 67 \ mod \ 120$. A solution $(u;\ v)$ of the equation $u^2 - 15Dv^2 = -20$ needs to be find. u will always be equal to $5 \ mod \ 15$. Thus, $x = (-5 \pm u)/15$. As in type F curves, p should be checked if it is a prime for x, and that $n = p(x)t(x) + 1$ has a large prime factor r. If yes, the complex multiplication equation is used to construct the curve.

3.1.3 Attribute-Based Encryption

Sahai and Waters subsequently introduced a new type of IBE called Fuzzy Identity-Based Encryption (FIBE) [71]. FIBE is closely related to the concept of IBE that was introduced by Shamir [77] and proposed by Boneh and Franklin [16] In FIBE, a private key is associated with a set of attributes, ω, and able to decrypt ciphertexts encrypted with a set of attributes, ω', if and only if at least k attributes overlap between ω' and ω. FIBE's motivation was to design an error-tolerant IBE that uses biometric identities as public keys. IBE and FIBE have limited applications, as they do not allow for a scalable and fine-grained access control to ciphertexts.

Bethencourt, Sahai, and Waters [13] constructed the first CP-ABE in which private keys are labeled with sets of attributes and ciphertexts are associated with access structures consisting of AND and OR gates. The limitation of this work

was the lack of expressing negative attributes due to the use of monotonic access structures. The definition of access structure is presented in Definition 3.1.4.

Definition 3.1.4. [Access Structure]
Let following be as set of parties: $\{P_1, P_2, ..., P_n\}$. *A collection* $\mathbb{A} \subseteq 2^{\{P_1, P_2, ..., P_n\}}$ *is monotone if* $\forall$ *B,C : if* $B \in \mathbb{A}$ *and* $B \subseteq C$ *then* $C \in \mathbb{A}$. *An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection)* $\mathbb{A}$ *of non-empty subsets of* $\{P_1, P_2, ..., P_n\}$, *i.e.,* $\mathbb{A} \subseteq 2^{\{P_1, P_2, ..., P_n\}} \setminus \{\varnothing\}$. *The sets in* $\mathbb{A}$ *are called the authorized sets, and the sets not in* $\mathbb{A}$ *are called the unauthorized sets.*

Current implementations of CP-ABE are typically based on the construction of a bilinear pairing between two elliptic curve groups [13].

A CP-ABE scheme consists of four fundamental algorithms: Setup, Encrypt, KeyGen, and Decrypt, and one optional algorithm, Delegate.

Bethencourt's CP-ABE Construction

Let $\mathbb{G}_0$ be a bilinear group of prime order p, and let g be a generator of $\mathbb{G}_0$. In addition, let e: $\mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ denote the bilinear map. A security parameter, k, will determine the size of the groups.

The Lagrange coefficient $\Delta_{i,S}$ for $i \in \mathbb{Z}_p$ and a set, S, of elements in $\mathbb{Z}_p$: $\Delta_{i,S}(x) = \Pi_{j \in S, j \neq i} \frac{x-j}{i-j}$ is defined. Additionally, a hash function H: $\{0,1\}^* \rightarrow \mathbb{G}_0$ is employed to map any attribute described as a binary string to a random group element.

- **Setup:**

  The setup algorithm begins with choosing a bilinear group $\mathbb{G}_0$ of prime order p with generator g. Next, the algorithm randomly selects two numbers: α and $\beta \in \mathbb{Z}_p$. Accordingly, the public key and master key are as follows.

$$PK = \mathbb{G}_0, g, h = g^{\beta}, f = g^{1/\beta}, e(g,g)^{\alpha}$$

$$MK = (\beta, g^{\alpha}),$$

where $\mathbb{G}_0$ is the bilinear group of prime order p, g is the generator of $\mathbb{G}_0$, $e(g, g)$ is the bilinear pairing, e: $\mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$.

- **KeyGen(*MK*, *S*):**

  The key generation algorithm takes two parameters, the master key MK of the CPABE scheme, and a set of attributes S that describe a subject, and outputs a private key for that subject SK. The algorithm first randomly selects a value $r \in \mathbb{Z}_p$, and then randomly chooses values $r_j \in \mathbb{Z}_p$ for each attribute $j \in S$. The secret key is solved as follows.

  $$SK = (D = g^{(\alpha+r)/\beta}, \forall\, j \in S\text{: } D_j = g^r \cdot H(j)^{rj}, D'_j = g_{rj}),$$

  where $H(j)$ is the hash function that relates any attribute described as binary string to an element of $\mathbb{G}_0$.

- **Encrypt(*PK*, *M*, $\mathcal{T}$):**

  The encryption algorithm takes three parameters, the public key PK, a message M, and a tree access structure $\mathcal{T}$ over a set of attributes. It will encrypt M and produce a ciphertext CT such that only a user who possesses the set of attributes satisfying the tree access structure will be able to decrypt CT. The encryption process starts with producing a polynomial q_x for each node x in $\mathcal{T}$ starting from the root node R. The degree d_x of these q_x are $d_x = k_x - 1$, where k_x is the threshold value of the node x. Starting with q_R ,the polynomial of root node R, the algorithm sets $q_R(0) = s$, where $s \in \mathbb{Z}_p$ is randomly chosen, and then chooses other random points to define the other coefficients of the polynomial q_R. For the rest of q_x of the other nodes x, the algorithm sets $q_x(0) = q_{parent(x)}\ (index(x))$ and chooses other randomly points to define the other coefficients of the polynomials q_x. After the polynomials are generated, the ciphertext is then constructed as follows.

  $$CT = (\mathcal{T}, \widetilde{C} = Me(g, g)^{\alpha s}, C = h^s,$$

$\forall\, y \in Y$: $C_y = g^{qy(0)}$, $C'_y = H(att(y))^{qy(0)})$, where Y is the set of all leaf nodes in $\mathcal{T}$.

- **Decrypt(*PK*, *CT*, *SK*):**

  The decryption algorithm takes three inputs, the public key *PK*, the ciphertext *CT*, which was obtained for a tree access structure $\mathcal{T}$, and the private key *SK* for a set *S* of attributes. If the set *S* of attributes satisfies the access structure $\mathcal{T}$, then the algorithm decrypts the ciphertext *CT* and returns the message *M*. The decryption algorithm performs the decryption in a two-step process. The first step determines if tree access structure $\mathcal{T}$ of the ciphertext *CT* satisfies the set of attributes *S* of *SK*. The second step decrypts *CT* to obtain *M*. The first step calls a recursive algorithm **DecryptNode(***CT***,***SK***,***x***)**, where x is a node from $\mathcal{T}$ starting from the root node *R*. If x is a leaf node, then $i = att(x)$. If $i \in S$, then

  DecryptNode(*CT***,***SK***,***x***)** $= \dfrac{e(D_i, C_x)}{e(D'_i, C'_x)}$

  $$= \frac{e(g^r \cdot H(i)^{ri}, g^{q_x(0)})}{e(g^{ri}, H(i)^{q_x(0)})}$$

  $$= e(g, g)^{rq_x(0)}.$$

  If $i \notin S$, then **DecryptNode(***CT***,***SK***,***x***)** $= \perp$.

  If x is not a leaf node, then **DecryptNode(***CT***,***SK***,***z***)** calls for all z where z are children of x, and stores the results in F_z. It then randomly chooses S_x a k_x-sized subset of child nodes of z where $F_z \neq \perp$. If S_x does not exist, then **DecryptNode(***CT***,***SK***,***x***)** $= \perp$; otherwise, a polynomial interpolation is performed on the nodes to compute $e(g, g)^{rq_z(0)}$. If the return value in both cases, x is a leaf node and a non leaf node, is not $\perp$ or an error value, the decryption process begins by calling the decryption algorithm starting from the root node *R* and setting

$$\text{A} = \textbf{DecryptNode(}CT\textbf{,}SK\textbf{,}R\textbf{)} = e(g,g)^{rq_R(0)} = e(g,g)^{rs}.$$

The decryption algorithm decrypts CT to obtain M by computing

$$\widetilde{C} \;/\; (e(C,D) \;/\; A) = \widetilde{C} \;/\; (e(h^s, g^{(\alpha+r)/\beta}) \;/\; e(g,g)^{rs}) = M.$$

- **Delegate(SK, $\widetilde{S}$):**

  The delegation algorithm takes as inputs a secret key SK for some set of attributes S and a set $\widetilde{S} \subseteq S$. It outputs a secret key $\widetilde{SK}$ for the set of attributes $\widetilde{S}$. As in the key generation algorithm, the delegation algorithm randomly selects a value $\widetilde{r}$ and $\widetilde{r_k}\ \forall\ k \in \widetilde{S}$. The delegate secret key $\widetilde{SK}$ is constructed as follows.

$$\widetilde{SK} = (\widetilde{D} = Df^{\widetilde{r}},\ \forall\ k \in \widetilde{S}\text{: } \widetilde{D_k} = D_k\, g^{\widetilde{r}}\, H(k)^{\widetilde{r}k},\ \widetilde{D'_k} = D'_k\, g_{\widetilde{r}k}).$$

Tools for implementing the CP-ABE scheme are the cpabe toolkit developed by Bethencourt et al. [32], the CHARM library developed by Akinyele et al. [1] and PIRATTE Jahid and Borisov [82]. The cpabe toolkit lacks the support of the delegation function and the ability to perform asymmetric bilinear pairings— such that e: $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. The CHARM library allows the construction of both symmetric and asymmetric bilinear pairings. On the other hand, the PIRATTE scheme requires the use of asymmetric pairings. This is to prevent subjects from maliciously combining the same group in case of using the symmetric pairings to skip the key revocation [46].

Martin [46] experimented with various types of curves for the different functions of CP-ABE. In **KeyGen(**MK**,**S**)** function, type F curves was the fastest, followed by A, D, E, and G curves. In the **Encrypt(*PK*, *M*, $\mathcal{T}$)** function, similar to the **KeyGen(**MK**,**S**)** function, type F curves was the fastest, then type A curves and type D curves. Type G curves was the slowest. In the **Decrypt(*PK*, *CT*, *SK*)**

function, type F curves performed the worst. Type A curves was the fastest, followed by type D and E curves. So in overall, the performance of type A, D, and F curves were acceptable for the usage of CPABE. However, type F curves may not be recommended for the poor performance in the **Decrypt(*PK*, *CT*, *SK*)** function.

3.2 A Cloud-Based EHR System

This section explores the proposed design, based on CP-ABE, for a cloud-based EHR system that provides secure EHR storage, with flexible, fine-grained access control.

3.2.1 The CP-ABE Scheme

Following Bethencourt, Sahai, and Waters [13], healthcare providers share one public key for encryption in the proposed CP-ABE design, thus avoiding PKI; however, each healthcare provider has a distinct secret key for decryption. CP-ABE supports complex policies to specify which secret keys can decrypt which ciphertexts: each healthcare provider's secret key is labeled with a set of attributes, and ciphertexts are associated with access policies. The secret key of a healthcare provider can decrypt a particular ciphertext only if the attribute set of the healthcare provider's key satisfies the access policy associated with that ciphertext, as illustrated in Figure 3.1. Here, the nurse practitioner with the ABCD Medical Group can access EHRs that are only allowed to physician assistants *or* nurse practitioners, *and* who work in the ABCD Medical Group; and the physician assistant with the WXYZ Medical Group is not allowed access.

CP-ABE thus supports flexible and fine-grained access control with healthcare providers being able to access only relevant EHRs encrypted with access policies that satisfy their keys' attributes. Also, if a secret key is compromised, only EHRs

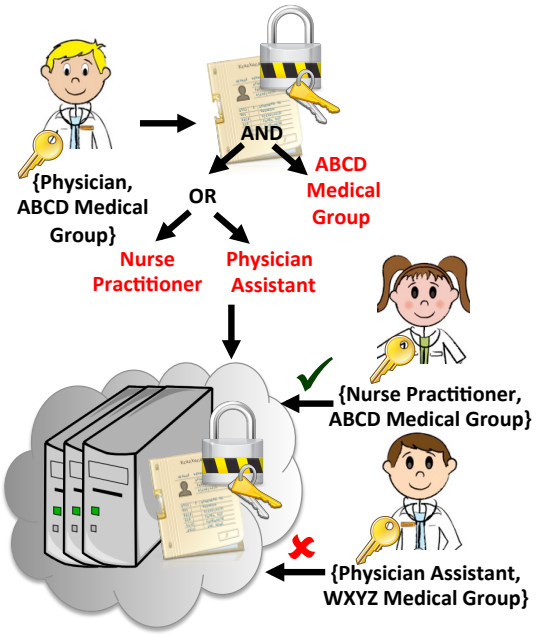

Figure 3.1: An example of using CP-ABE in a cloud-based EHR system. A record is encrypted with an access structure stating that any physician assistant or nurse practitioner who works in the ABCD Medical Group can decrypt the record. Here, the nurse practitioner is allowed to access (and decrypt) the record, but not the physician assistant as he doesn't work in the ABCD Medical Group.

that can be decrypted with that key will be compromised; other EHRs are still protected.

3.2.2 System Architecture

In the proposed scheme, EHRs are stored in the cloud, and can be accessed through a web portal by multiple users who are typically healthcare professionals. The users who create EHRs are also responsible for generating access policies based on the attributes of authorized healthcare providers, encrypting EHRs based on the generated policies and uploading encrypted EHRs into the cloud. EHRs are organized into a labeled hierarchical data structure [10], which makes it possible

to share different parts of the EHR, thus making the scheme more flexible.

Figure 3.2 shows the architecture for the proposed cloud-based EHR system, which consists of three main components: the cloud-based EHR system, healthcare providers (users), and the Attribute Authority (AA). The system uses two fundamental cloud services: data storage and computing resources. The first service is for storing encrypted EHRs that are accessible only to healthcare providers through authentication mechanisms, and access policies based on complete attributes of healthcare providers. The second service is for hosting the web portal, generating access policies, and performing other needed computing tasks.

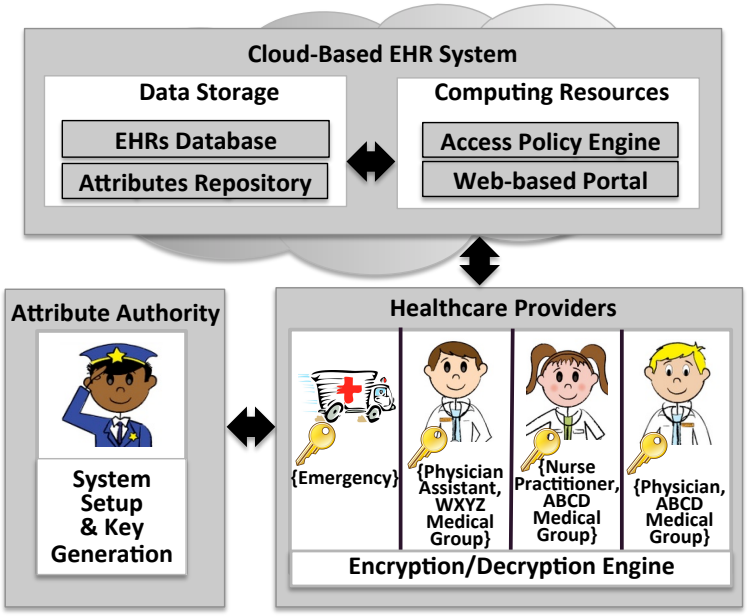

Figure 3.2: The cloud-based EHR system architecture consisting of three main components: the cloud-based EHR system, healthcare providers (users), and the attribute authority (AA).

Once healthcare providers obtain their private keys from the AA, they log in to the system using their username and password; on first login, they will need to download and install lightweight software for encrypting and decrypting EHRs

locally. When a healthcare provider requests access to an encrypted record, she will first locate and download it, and then use her key and the lightweight software to decrypt it. To upload a new record, she will first request the desired attributes and generate the access policy using the Access Policy Engine; encrypt the record using the lightweight software; and finally upload the encrypted record.

3.2.3 Key Management

Key management must be cost-effective and its components are carefully implemented. In the proposed system, key management is handled appropriately for cloud-based EHR systems.

- **Generation and Distribution:** The AA generates the public and master private keys using the Setup algorithm. When a new healthcare provider joins the system, the AA derives a distinct secret key associated with her attributes by running the Key Generation algorithm off-line. Healthcare providers must store their secret keys securely, and regenerate keys after a predefined expiration date. Secret key regeneration is performed without the need to refresh the system parameters, public key, and master private key. Healthcare providers can only seek their secret keys through their healthcare organizations.

- **Revocation:** The cloud-based EHR requires forward secrecy such that when a healthcare provider's access is revoked, she should not be allowed to access EHRs that she was able to access before being revoked. Owners of encrypted EHRs have the option to add an expiration date to access policies used for encryption, or to re-encrypt them with updated access policies to prevent access by revoked healthcare providers. The problem of re-distributing secret keys is thus avoided.

- **Escrow:** In the cloud-based EHR system, the AA can regenerate secret keys for healthcare providers to access EHRs during emergencies.

3.3 Analysis

To evaluate the feasibility of the proposed design, several experiments were conducted to measure overhead in terms of time and storage. The experiments were run on a virtual machine with 1GB of RAM, and hosted on 2.26GHz. The CP-ABE implementation uses a 160-bit elliptic curve group on the curve $y^2 = x^3 + x$ over a 512-bit finite field. It also uses the Pairing Based Cryptography library [43] to perform the pairing-based cryptosystems utilizing bilinear mapping.

3.3.1 Time Overhead

To measure the efficiency of the encryption and decryption algorithms, five image files (1MB, 10MB, 20MB, 30MB, 40MB, and 50MB) were encrypted with six different numbers of attributes in the access structure, and then decrypted with a secret key that is associated with ten attributes. Figure 3.4 shows how the encryption time increases almost linearly with the number of attributes in the access structures. This is because when files are encrypted, the access trees need to be created over the set of attributes, and the more number of attributes, a single access tree has, the more polynomials need to be produced. On the other hand, Figure 3.3 shows that the decryption time tends to be independent of the number of attributes. This is because a single key is used, and decryption time is spent on verifying whether the access structure associated with the ciphertext satisfies the attributes of the single key. Each experiment was repeated five times, and the elapsed time was averaged to yield the efficiency measure.

In the context of EHR systems, records less than 1MB are encrypted in less

than 0.2 seconds depending on the number of attributes in the access structures, and decrypted in less than 0.1 seconds, as shown in figures 3.4 and 3.3. On the other hand, medical images that are 30MB are likely to be encrypted in less than 0.5 seconds with respect to the number of attributes in the access structures, and decrypted in less than 0.4 seconds. These results indicate that the time performance of CP-ABE is feasible for EHR systems.

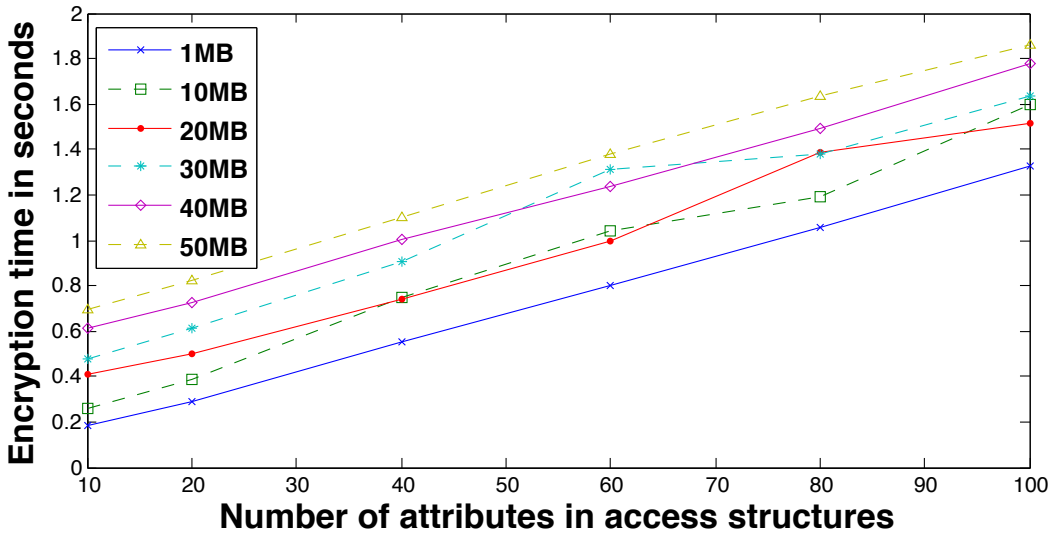

Figure 3.3: Encryption time for varying number of attributes in the access structure (for five image sizes). Encryption time increases almost linearly with the number of attributes in the access structures as access trees need to be created.

3.3.2 Storage Overhead

To determine storage overhead, the difference between encrypted records and decrypted records across various numbers of attributes in the access structures was measured. The difference is relatively small for all the files and increases linearly

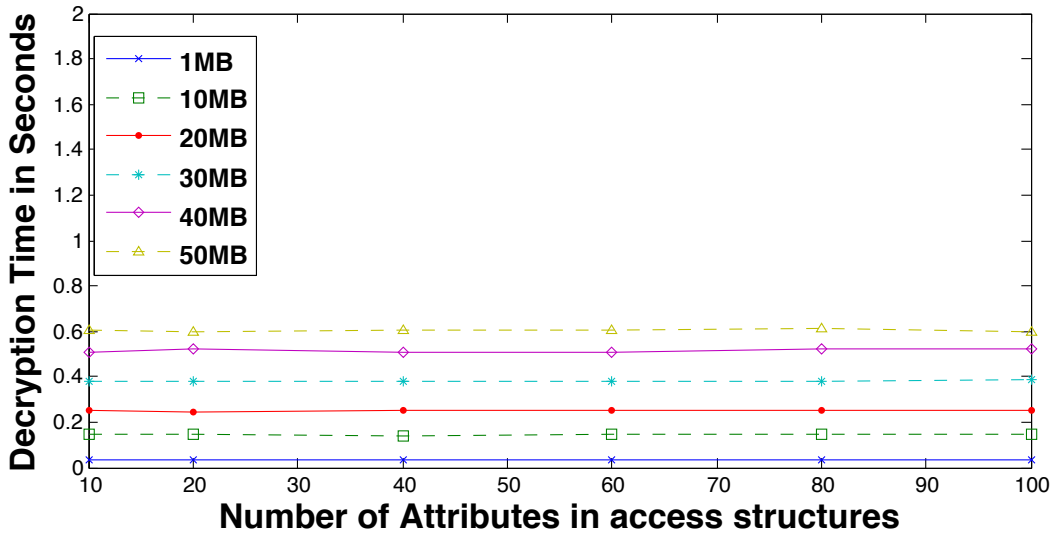

Figure 3.4: Decryption time for varying number of attributes in the access structure (for five image sizes). Decryption time tends to be independent of the number of attributes as a single key is used.

with respect to the number of attributes in the access structures as shown in Figure 3.5. This is also because when files are encrypted, the created access trees over the set of attributes need to be stored, and the more number of attributes a single access tree has, the more polynomials need to be produced and stored.

The difference is less than 4KB for simpler access structures and less than 30KB for complex access structures. The results suggest that storage overhead is negligible in the context of cloud-based EHR systems because cloud service providers currently (and will continue to) provide large amounts of storage at reasonable prices.

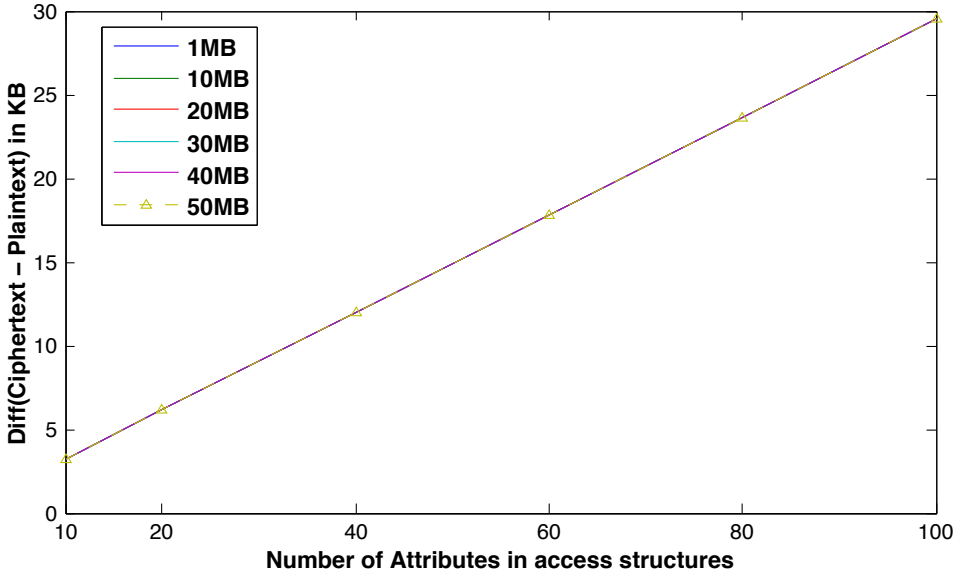

Figure 3.5: Difference between ciphertext and plaintext in size for varying number of attributes in the access structure (for five image sizes). The difference increases linearly with the number of attributes in the access structures due to the need to store the generated access trees.

3.4 Related Work

In this chapter, the terms electronic medical record (EMR), electronic health record (EHR), and personal health record (PHR) systems followed the definitions provided by the National Alliance for Health Information Technology [57].

EMR systems manage EHRs that can be created, gathered, and managed by authorized healthcare providers *within one healthcare organization*. Benaloh et al. [10] propose a patient controlled encryption EMR system that enables patients to mediate access control decisions over their medical records. EHRs are partitioned into a hierarchical structure in which each section is encrypted with a derived public key that patients are required to manage, and decrypted with a derived subkey

from a master private key. This system has several issues: potential key management overhead, no support for a key escrow agent in emergencies, and likely data integrity issues as EHRs are managed by patients, not healthcare providers.

EHR systems manage EHRs that can be created, gathered, and managed by authorized healthcare providers across *more than one* healthcare organization. Although several EMR and PHR systems have been proposed, there is little research in the area of cloud-based EHR systems. Commercial cloud-based EHR systems such as Practice Fusion [64] and CareCloud [17] are available; although they are HIPAA compliant, the security of EHRs is only based on access control decisions mediated by their cloud storage providers. EHRs are stored unencrypted, which has security and privacy implications.

PHR systems manage personal EHRs that are *imported* from EMR and EHR systems by patients. Narayan et al. [56] introduce a patient controlled encryption PHR system that enables a patient to import EHRs into the cloud, and encrypt each record with SKE using different keys. Along with each encrypted record, the patient uploads a corresponding entry consisting of encrypted metadata using a broadcast ABE, unencrypted access policy, and a search-index. For healthcare providers to access a record, they must decrypt the metadata entry to find the location and name of the record and the symmetric key. They then request the encrypted record from the cloud server, and decrypt it using the symmetric key. In Narayan's scheme, only patients can be the owners of EHRs, thus this scheme cannot be used for cloud-based EHR systems. HealthVault [49], a cloud-based PHR service offered by Microsoft, and the soon-to-be discontinued Google Health [25] service both guarantee encryption during EHR transmission from patient to the cloud and back, but to secure EHR storage, they seem to rely primarily on proper access control and limiting physical access rather than full encryption.

Finally, substantial attention has been paid recently to querying encrypted data

stored externally or in the cloud, e.g., Li at al. [41], but the focus here is on efficient and effective access control of cloud data.

3.5 Chapter Summary

This chapter presented a general design for a secure cloud-based system using CP-ABE and applied it to EHRs. This approach provided an effective solution to several issues related to standard encryption and access control mechanisms. It also investigated the feasibility of adopting CP-ABE in terms of performance and storage overhead. The results suggest that the proposed design would provide promising performance and consume negligible storage, and thus CP-ABE can be used to enforce access control in information-sharing environments.

Chapter 4

The BiLayer Access Control Model (BLAC)

This chapter addresses a major objective of this dissertation which is the exploration of using attributes and policies for developing effective access control models. It describes the work toward this objective in addressing the problem of advancing access control models and mechanisms to support the use of attributes and access-control policies while preserving the advantages of RBAC. The result has been the design and validation of BLAC model [7], which was in part inspired by aspects of CP-ABE presented in Chapter 3.

The rest of this chapter defines the BLAC model, describes the concept of *pseudoroles* and their generation, and presents policy specification and evaluation methods used in the BLAC model. It then informally validates the practical viability of BLAC model by applying it to the healthcare domain.

4.1 Overview of the BLAC Model

The BLAC model is designed to support attributes and policies while preserving the advantages of RBAC in terms of access control administration and user permission review. The model uses the concept of *pseudorole*, which is informally defined as a set of values of static attributes of subjects. A pseudorole is not a real role, which is traditionally defined as a job function in RBAC. This is because roles in RBAC are defined independently from subject attributes.

Typically, subjects' attributes are categorized as *static* (when attribute values typically do not change) and *dynamic* (when attribute values change frequently). Static attributes are for generating *pseudoroles*. Static and dynamic attributes are used in policies to constrain *pseudoroles*.

The main concepts of BLAC are:

1. Subjects are assigned to *pseudoroles* and objects are associated with *BLAC policies*; these terms are formally defined in Definition 4.1.1 and Definition 4.1.2.

2. Access control is performed on a two-step evaluation procedure: (1) PseudoRole function evaluation, and (2) rule(s) evaluation.

Definition 4.1.1. [Pseudorole]
Let $s \in S$ be associated with a set of attributes $sat \subseteq SAT$. A pseudorole $pr \in PR$ of s is a collection of $attrValue_i$ of $attr_i$ such that $attr_i \in sat$ and $attr_i$ is a static attributes of s.

Definition 4.1.2. [BLAC Policy]
A BLAC policy is a five-tuple, $p = \langle pr, sat, obat, eat, opat \rangle$, where $pr \in PR$, $sat \subseteq SAT$, $obat \subseteq OBAT$, $eat \subseteq EAT$, and $opat \subseteq OPAT$.

The BLAC model includes four basic data sets called S (subjects), OB (objects), PR (pseudoroles), P (policies). The BLAC model is fundamentally defined as assigning subjects to pseudoroles and associating objects to policies. Thus, SPR (subject-to-pseudorole assignment) and OBP (object-to-policy assignment) are two relations that relate subjects to pseudoroles and objects to policies respectively. *Assigned_subject* maps a subject to a pseudorole, and *assigned_object* maps an object to a policy.

Subjects and objects in the BLAC model are defined consistently with earlier access control models [23]. A subject is defined as an entity that requires access to system resources (e.g. human users, machines or programs). An object is defined as a system resource to which access must be controlled (e.g. files and directories, columns, rows, and tables or printers).

The BLAC model is formally defined as follows:

- $S = \{s_1, \ldots, s_n\}$ is a set of subjects. Each subject has a unique identifier;
- $SAT = \{sat_1, \ldots, sat_n\}$ is a set of subject attributes, where $sat_i = \{a_{i1}, \ldots, a_{im}\}$ is the attributes of subject $i \in S$. Each attribute is a $< attribute : attributeValue >$ pair, $a_{ij} =< attr_j : attrValue_j >$, where $attr_j$ is an attribute identifier and $attrValue_j$ is an attribute value of this identifier for subject i;
- $OB = \{ob_1, \ldots, ob_n\}$ is a set of objects. Each object has a unique identifier;
- $ObAT = \{obat_1, \ldots, obat_n\}$ is a set of object attributes. $obat_i = \{a_{i1}, \ldots, a_{im}\}$ is the attributes of object $i \in Ob$. Each attribute entry is a $< attribute : attributeValue >$ pair, $a_{ij} =< attr_j : attrValue_j >$, where $attr_j$ is an attribute identifier and $attrValue_j$ is an attribute value of this identifier for object i;
- $EAT = \{eat_1, \ldots, eat_n\}$ is a set of environment attributes. Each attribute entry is a $< attribute : attributeValue >$ pair, $eat_i =< attr_i : attrValue_i >$, where $attr_i$ is

an attribute identifier and $attrValue_i$ is an attribute value of this identifier;

- $OpAT = \{opat_1, \ldots, opat_n\}$ is a set of operation attributes. Each attribute is a $< attribute : attributeValue >$ pair, $opat_i =< attr_i : attrValue_i >$, where $attr_i$ is an attribute identifier and $attrValue_i$ is an attribute value of this identifier;
- $PR = \{pr_1, \ldots, pr_n\}$ is a set of pseudoroles as defined in Definition 4.1.1;
- $P = \{p_1, \ldots, p_n\}$ is a set of BLAC policies as defined in Definition 4.1.2;
- $SPR \subseteq S \times PR$, a total function, subject-to-pseudorole assignment, that relates $s \in S \mapsto pr \in PR$;
- $OBP \subseteq OB \times P$, a total function, object-to-policy assignment, that relates $ob \in OB \mapsto p \in P$;
- $assigned\_subject : S \mapsto PS$, the mapping of a subject $s \in S$ to exactly one pseudorole $pr \in PR$;
- $assigned\_object : OB \mapsto P$, the mapping of an object $ob \in OB$ to exactly one policy $p \in P$;

Figure 4.1 depicts the BLAC model.

4.2 Pseudorole Generation

In the BLAC model, pseudoroles are the values of static attributes of subjects. Static attributes used to compose pseudoroles are determined by humans to ensure the rare changes. The data set of pseudoroles can be generated manually from pairing all the values of the candidate static attributes. However, an automatic approach borrowed from role mining in RBAC systems can be applied for generating pseudoroles to accelerate the process [100]. The approach of pseudorole generation is

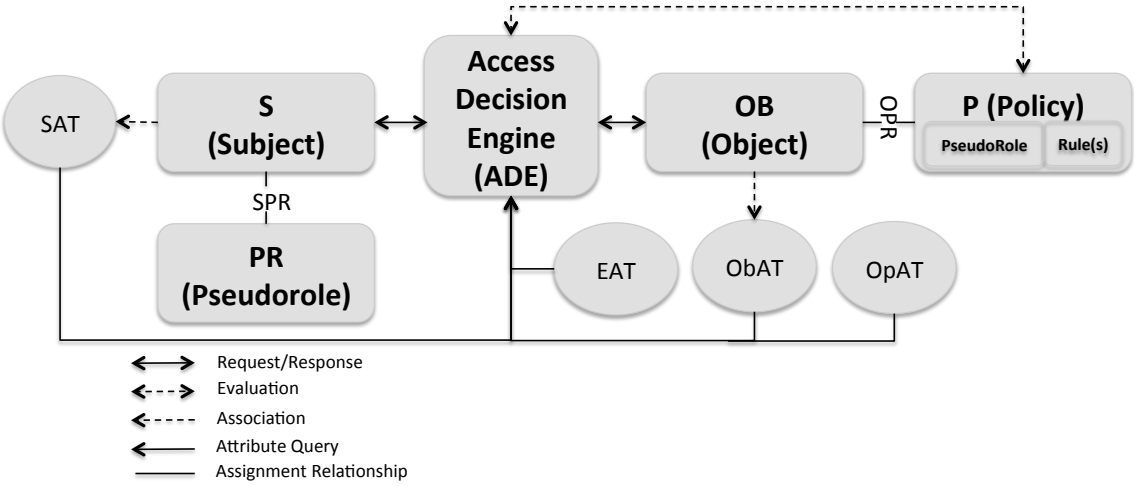

Figure 4.1: The components of the BLAC model containing four data sets: subject, pseudorole, object and policy, and two relations: subject-pseudorole assignment and object-policy assignment.

yet different from role mining approaches, in which role mining uses permissions possessed by all subjects to generate roles. In the BLAC model, the values of the attributes associated with all subjects are used to generate pseudoroles. The following paragraphs describe the pseudorole generating approach.

First, all distinct values in the first attribute column are found. Each value is considered as a root of a tree. Then, all distinct values of the second attribute column are found and added as nodes to each rooted tree. The process continues depending on the number of attributes used to generate pseudoroles. Thus, if three attributes are used, the process is repeated three times. After creating the trees, each path from a leaf to the root is considered as a pseudorole.

To demonstrate the pseudorole generating approach, assume a hypothetical medical center with nine subjects. Each subject is defined by *name*, identification number *ID*, *gender*, the field of the healthcare *provider*, their *department*, and their office *location*. The nine subjects are presented in Table 4.1. *Provider*, *Department*, and

Location are chosen for generating pseudoroles for being changed less frequently at high granularity. Based on this example, the complete set of generated pseudoroles, as shown in Figure 4.2, is four trees with 24 distinct pseudoroles. In the first tree, the path from the root to the leaf is "physician," OB/GYN" and "A." Thus, "Physician OB/GYN A" is a pseudorole.

All the generated pseudoroles in a system could considered even though not all the pseudoroles can currently be associated with subjects because new subjects might be added later to these unused pseudoroles. However, unreasonable pseudoroles such as "Physician Billing A" and "Physician Billing B" can be eliminated. Although "Physician OB/GYN A" is a pseudorole, it is still three distinct attribute values.

Table 4.1: Attributes of subjects demonstrating the pseudorole generating approach, showing nine subjects with six attributes. Provider, Department and Location are chosen for generating pseudoroles for being changed less frequently at high granularity.

Name	**ID**	**Gender**	**Provider**	**Department**	**Location**
E. Robert	345-765	Female	Physician	OB/GYN	A
A. Mark	526-874	Male	Physician	OB/GYN	A
H. John	231-938	Female	Nurse	OB/GYN	A
M. Martin	657-923	Female	Administrative Staff	OB/GYN	A
E. Arthur	112-681	Male	Billing Staff	Billing	A
J. Fox	437-348	Male	Physician	PCP	B
H. Anderson	256-828	Female	Nurse	PCP	B
F. Brown	562-910	Female	Administrative Staff	PCP	B
D. Lee	102-581	Male	Billing Staff	Billing	B

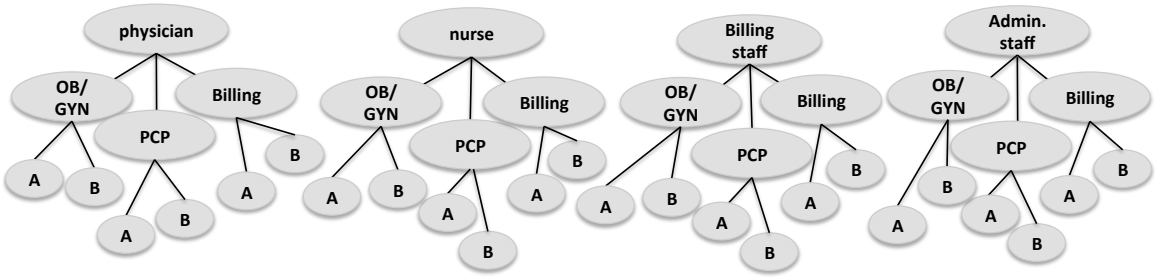

Figure 4.2: The complete set of generated pseudoroles. A pseudorole in here is the path from a root to a leaf. For example, in the first tree, the path from the root to the leaf is "physician," OB/GYN" and "A." Thus, "Physician OB/GYN A" is a pseudorole.

4.3 BLAC Policy Specification and Evaluation

To regulate access by subjects performing operations over objects within certain environment settings via the BLAC model, it is essential to compose a policy structure based on the proposed BLAC model. A BLAC policy is specified to enable the two-step evaluation procedure performed by the BLAC model, as discussed in the following paragraphs.

A BLAC policy consists of two elements: a boolean function called *PseudoRole*, and a set of one or more rules defined as boolean functions. Each rule has four sub-elements also defined as boolean functions specifying the range of values that must be satisfied for the subject, object, operation, and environment attributes. A policy structure (using the XACML policy format [89]) is shown in Figure 4.3.

The PseudoRole function performs the first-step access evaluation by verifying whether a subject requesting access to an object holds the right pseudorole specified by the BLAC policy. Each rule has four sub-elements defined as boolean functions specifying the range of values that must be satisfied in terms of the attributes of subject, object, operation, and environment. The rules are for performing the second-step access evaluation.

```
POLICY STRUCTURE
<Policy>
    <PseudoRole>...</PseudoRole>
    <Rule>
      <Subject>...</Subject>
      <Object>...</Object>
      <Operation>...</Operation>
      <Environment>...</Environment>
    </Rule>
    <!- more rules can be specified
</policy>
```

Figure 4.3: The structure of a BLAC policy, showing the two elements: PseudoRole boolean function, and a set of one or more rules defined as boolean functions that specify the range of values that must be satisfied for the subject, object, operation, and environment attributes.

Using the above-mentioned policy elements, a BLAC policy specification is defined as follows:

Definition 4.3.1. [BLAC Policy Specification]

Let $p \in P$ be a BLAC policy that determines whether s can access o. The BLAC policy specification is defined as $p = \{PRF, Ru_1, \ldots, Ru_n\}$, such that PRF is PseudoRole boolean function and $Ru_1, \ldots, Ru_n$ are a finite set of access rules. Each $Ru_i = \{RF_1, \ldots, RF_n\}$, such that RF_i is a set of boolean functions.

To evaluate an access request, the two-step evaluation procedure is performed by the access decision engine (ADE) in the BLAC model. In the first step, ADE checks the BLAC policy associated with the requested object to verify whether the requesting subject has the pseudorole specified in the BLAC policy or not. If the requesting subject holds the right pseudorole, ADE in the second step checks the rule(s) within the associated BLAC policy for additional constraints to grant or deny the access request. Figure 4.4 illustrates the evaluation procedure of BLAC policies by ADE in the BLAC model.

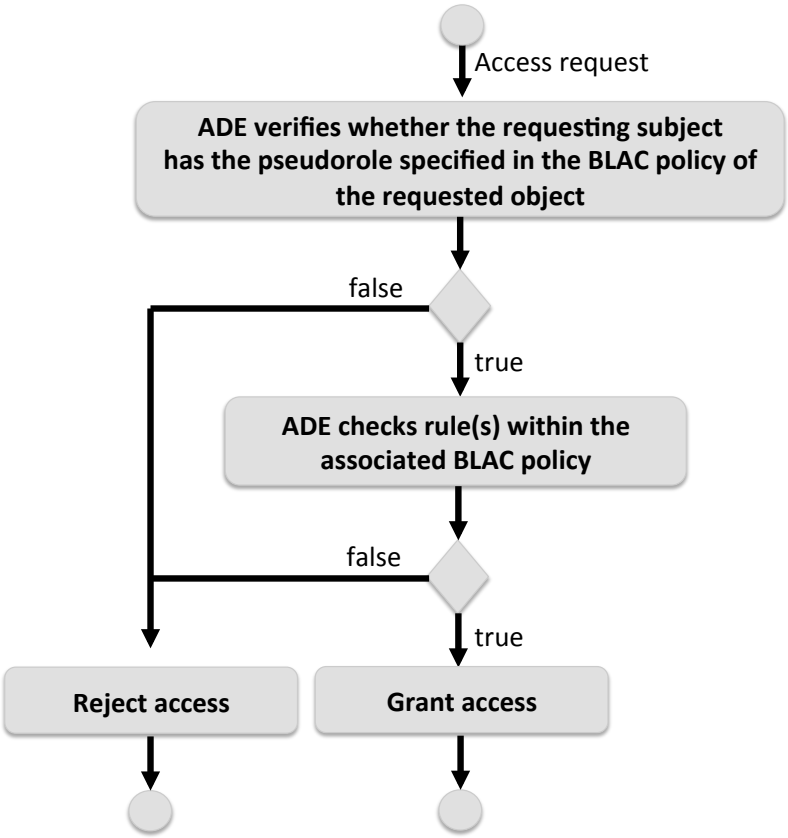

Figure 4.4: The two-step evaluation procedure for BLAC policies. First, ADE checks the BLAC policy associated with the requested object to verify whether the requesting subject has the pseudorole specified in the BLAC policy or not. If the requesting subject holds the right pseudorole, ADE in the second step checks the rule(s) within the associated BLAC policy for additional constraints to grant or deny the access request.

4.4 Informal Validation of BLAC

This section informally validates the practical viability of the BLAC model by applying it to the healthcare domain. This validation is needed to ensure that the BLAC model can fulfill its intended main objective mentioned in Chapter 1, that is, attributes can be used to create an effective access control model while preserving the advantages of RBAC.

This informal validation of the BLAC model examines each of the core functions of access control identified in Chapter 5 and summarized in Table 5.1. To

demonstrate these functions, a usage scenario from the relevant healthcare literature [81, 97] is adopted to define a set of use cases. These use cases are then used to show that the BLAC model can effectively perform the identified functions.

The use case terminology used here is based on the pattern specifically applied to access control, both by Oracle [48] and Amazon Web Services [8]. In this approach, a *scenario* describes a narrative of common, previously-known interactions between users and the software system, and *use cases* are common functions supported by the system for users. Note that the use case approach is substantially different from the Unified Modeling Language approach to *use cases* [24].

The rest of this section describes a typical scenario, as depicted in Figure 4.5, in the healthcare world that makes use of an EHR system. This scenario is used by van der Linden et al. [97] to illustrate the sharing of EHRs. The scenario adapted in this dissertation describes a sequence of actions that require access to information in an EHR system. Using the BLAC model, such accesses are intercepted and examined to enforce access control policies specified by a healthcare organization. The scenario, however, is slightly adjusted to demonstrate the core functions of access control, and thus, some examples from Silow-Carroll et al. [81] are adopted.

In the scenario shown in Figure 4.5, five use cases are defined to describe the complete life-cycle of any access control model including the BLAC model: (1) subjects join a system, (2) subjects request access, (3) the organization's policies are modified, (4) subjects' permissions are modified, and (5) subjects leave the system. Each of these use cases is detailed in the following sections. Each section describes a use case and then shows how the BLAC model implements the use case.

4.4.1 Use Case 1: Subjects Join a System

This section first describes the use case where subjects join a system, and then illustrates how the BLAC model carries out the use case.

A patient named Bob lives in a town with a large hospital, ABCA, and several general practitioners practices that share relevant health information via an EHR system. Bob has one specific primary care physician (PCP) whom he regularly visits and who is fully informed about Bob's medical history. Bob visits his PCP for a rash later. The PCP then decides to refer him to the dermatology department at the hospital for an expert consultation. Bob visits the dermatologist, who requests an access to his health information to know about Bob's allergies and medication. The dermatologist prescribes tablets to treat the rash. Later on, the hospital implements a new feature in the EHR system allowing only physicians to access the EHR system remotely. A month later, the dermatologist has been promoted to chair of the dermatology department. One day, Bob receives a mail stating that his PCP is leaving the practice and Bob is assigned to a new PCP.

Figure 4.5: Healthcare Scenario. This scenario is used to describe the five use cases to help validate the practical viability of the BLAC model.

Overview:

From the scenario (as shown in Figure 4.5), the PCP and dermatologist are hired and are provided access to their patients' health information via the EHR system. The PCP can access all of Bob's health records. The dermatologist can access Bob's health records that are related to the dermatology department. Additionally, when Bob joins the EHR system, he is permitted to access only his health information.

Implementation:

Upon employment, the PCP and dermatologist are assigned appropriate subject attributes. The subject attributes of the PCP and dermatologist are described in Table 4.2. They are also assigned pseudoroles in a table that represents the relation *SPR*, review the definition of the BLAC model in 4.1, in the BLAC model, as shown in Table 4.3. In this example, "Provider," "Department" and "Hospital Affiliation" are used for generating pseudoroles. Bob's attributes are provisioned upon his first visit to the practice; his attributes are shown in Table 4.4.

Table 4.2: Attributes of the PCP and dermatologist. These attributes are assigned upon their employment.

Attributes	**PCP**	**Dermatologist**
ID	345-765	231-938
First Name	Robert	Lisa
Last Name	Martin	John
DoB	May 8, 1969	Jan 31, 1964
Gender	Male	Female
Provider	Physician	Physician
Job Title	Physician	Physician
Department	Primary Care	Dermatology
Specialty	Internal Medicine	Dermatopathology
Graduation Year	1995	1991
Board Certification	Yes	Yes
Hospital Affiliation	ABCA	ABCA
Office Address	3 North Rd.	3 North Rd.
Office Phone	562-6813	638-8735
Office Number	1-581	3-731

Table 4.3: The table that holds the assignment of subjects to pseudoroles. This table shows the assignments of PCP and dermatologist to their pseudoroles. The attributes that compose the pseudorole in here are: provider, department and hospital affiliation.

Subjects	**Pseudoroles**
345-765	Physician,Primary Care,ABCA
231-938	Physician,Dermatology,ABCA

Table 4.4: Attributes of the patient Bob. These attributes are assigned upon his first visit to the practice.

Attributes	**Bob**
Name	Bob Lee
MRN	7659834
DoB	Jun 8, 1979
Gender	Male
ProviderID	345-765

```
POLICY 1
<Policy>
    <PseudoRole>
      <(subject.provider="physician"∨subject.provider="nurse")∧
       subject.provider="primaryCare"∧
       subject.hospital ="abca">
    </PseudoRole>
    <Rule>
      <Subject>"any"</Subject>
      <Object><object.providerID=subject.ID></Object>
      <Operation><action.type="read"∨action.type="modify"></Operation>
      <Environment><environment. AccessIP="192.123.*.*"></Environment>
    </Rule>
</policy>
```

Figure 4.6: Policy 1. Bob's EHRs created by his PCP are associated with policy 1, which allows subjects who hold pseudoroles with the values physician or nurse, primaryCare, and abca in provider, department, and hospital affiliation respectively. Policy 1 has one rule allowing any physician or nurse who works for the primary care department in the ABCA hospital to only read and modify records of their patients as long as the access request is issued within the hospital's network.

Upon the creation of Bob's EHRs, object attributes are assigned and access policies are specified and associated to these records. Access policies are stored, not bound to the EHRs, and referenced in a table that represents the relation *OBP* in the BLAC model, review the definition of the BLAC model in 4.1. EHRs are defined in the system using four object attributes: patient name, patient MRN, patient DOB, and the ID of the provider responsible for treating the patient. Bob's EHRs created by his PCP are associated with Policy 1 shown in Figure 4.6. However, Bob's EHRs created by the dermatology department are associated with Policy 2 shown in Figure 4.7.

4.4.2 Use Case 2: Subjects Request Access

This section describes the use case where subjects request access to objects, and then shows how the BLAC model carries out the use case.

```
POLICY 2
<Policy>
    <PseudoRole>
      <(subject.provider="physician"∨subject.provider="nurse")∧
       (subject.department="dermatology"∨ subject.provider="primaryCare")∧
       subject.hospital ="abca">
    </PseudoRole>
    <Rule>
      <Subject>"any"</Subject>
      <Object><object.providerID=subject.ID></Object>
      <Operation><action.type="read"∨action.type="modify"></Operation>
      <Environment><environment. AccessIP="192.123.*.*"></Environment>
    </Rule>
</policy>
```

Figure 4.7: Policy 2. Bob's EHRs created by the dermatology department are associated with policy 2, which allows subjects who hold pseudoroles with the values physician or nurse, dermatology or primaryCare, and abca in provider, department, and hospital affiliation respectively. Policy 2 has one rule allowing any physician or nurse who works for the dermatology or primary care departments in the ABCA hospital to only read and modify records of their patients as long as the access request is issued within the hospital's network.

Overview:

In the scenario shown in Figure 4.5, the dermatologist requests access to Bob's health information to check his current allergies and medication when Bob visits his practice. Such information is found in the patient demographics record that are accessible to all physicians and nurses.

Implementation:

The dermatologist requests access to Bob's demographics record that stores information about Bob's current allergies and medication. This record is associated with Policy 3 shown in Figure 4.8.

When the dermatologist issues the access request, the ADE in the BLAC model checks the policy associated with Bob's demographics record, which is Policy 3. According to Policy 3 and Table 4.3, the pseudorole of the dermatologist satisfies

```
POLICY 3
<Policy>
    <PseudoRole>
      <(subject.provider="physician"∨subject.provider="nurse")∧
       subject.department="any"∧
       subject.hospital ="abca">
    </PseudoRole>
    <Rule>
      <Subject>"any"</Subject>
      <Object>"any"</Object>
      <Operation><action.type="read"∨action.type="modify"></Operation>
      <Environment><environment.AccessIP="192.123.*.*"></Environment>
    </Rule>
</policy>
```

Figure 4.8: Policy 3. Bob's demographics record is associated with policy 3, which allows subjects who hold pseudoroles with the values physician or nurse, any, and abca in provider, department, and hospital affiliation respectively. Policy 3 has one rule allowing any physician or nurse who works for any department in the ABCA hospital to only read and modify this record as long as the access request is issued within the hospital's network.

the PseudoRole boolean function in Policy 3. Thus, the rule is further checked, and the access request is granted. The dermatologist can now read and modify Bob's demographics record.

4.4.3 Use Case 3: Organizational Policies are Modified

This section describes the third use case, that is the organization modifies one or more of its access policies. The section then shows how the BLAC model carries out the use case.

Overview:

In the scenario, the hospital implements a new feature in the EHR system allowing only physicians to access the EHR system remotely. Accordingly, a new access rule stating that *"Physicians are allowed to remotely access EHRs for patients who are under their responsibility"* is specified and enforced.

Implementation:

Changes in access policies made by the hospital are handled by updating the relevant policies. Thus, in this use case, system or security administrators use any policy editor tool to modify all the policies in the system to add the new access rule. For example, Policy 2 is modified by adding a new rule (the second rule) to state that *"Physicians are allowed to remotely access EHRs for patients who are under their responsibility"* as in Policy 4 shown in Figure 4.9.

```
POLICY 4
<Policy>
    <PseudoRole>
      <(subject.provider="physician"∨subject.provider="nurse")∧
       (subject.department="dermatology"∨ subject.department="primaryCare")∧
subject.hospital ="abca">
    </PseudoRole>
    <Rule>
      <Subject><subject.provider="physician"></Subject>
      <Object><object.providerID=subject.ID></Object>
      <Operation><action.type="read"∨action.type="modify"></Operation>
      <Environment><environment.AccessIP="any"></Environment>
    </Rule>
    <Rule>
      <Subject>"any"</Subject>
      <Object><object.providerID=subject.ID></Object>
      <Operation><action.type="read"∨action.type="modify"></Operation>
      <Environment><environment.AccessIP="192.123.*.*"></Environment>
    </Rule>
</policy>
```

Figure 4.9: Policy 4. After the hospital has changed one of its access policies allowing only physicians to remotely access the EHR system, BLAC policies need to be updated. Foe example, Policy 2 is modified in Policy 4 to include a new rule allowing only physicians to remotely access (to only read and modify) their patients records via the EHR system.

4.4.4 Use Case 4: Subjects' Permissions are Modified

This section describes the fourth use case, that is permissions of subjects are modified. The section then shows how the BLAC model carries out the use case.

Table 4.5: The implementation of changing the dermatologist's permissions, and since in BLAC, subject's permissions stem from his or her attributes, the implementation requires only a simple modification to the value of the dermatologist's attribute, "Job Title". The change is from "Job Title = Physician" to "Job Title = Chair".

(a) The value of the dermatologist's attribute before the change.

Attributes	**PCP**	**Dermatologist**
Job Title	Physician	Physician

(b) The value of the dermatologist's attribute after the change.

Attributes	**PCP**	**Dermatologist**
Job Title	Physician	Chair

Overview:

From the scenario, the dermatologist has been promoted to chair of the dermatology department, thus, his new job title is *"Chair"*. Hence, he is given administrative and financial duties that require access to information beyond medical records. Therefore, his set of permissions need to be updated.

Implementation:

The new change in the dermatologist's permission requires a simple modification to the value of the attribute, *"Job Title"*. The change is from *"Job Title = Physician"* to *"Job Title = Chair"*. No adjustments to existing access policies or rules are needed to grant the obligatory access. Additionally, no modifications to subject membership into pseudoroles are needed in the *SPR* relation, according to the definition of the BLAC model in 4.1. The new change to the attribute job title is shown in Table 4.5.

4.4.5 Use Case 5: Subjects leave a System

This section describes the fifth use case, that is subjects leave a system. The section then shows how the BLAC model carries out the use case.

Overview:

In the scenario, the PCP leaves the practice, and thus the PCP can no longer access Bob's health records. Bob is assigned to a new PCP. Bob's new PCP can access all his health records.

Table 4.6: The implementation of revoking the PCP by deleting the assignment of the PCP to his pseudorole in the assignment table.

(a) The assignment table before revoking the PCP.

Subjects	**Pseudoroles**
345-765	Physician,Primary Care,ABCA
231-938	Physician,Dermatology,ABCA

(b) The assignment table after revoking the PCP.

Subjects	**Pseudoroles**
231-938	Physician,Dermatology,ABCA

Implementation:

When the PCP leaves the practice, he immediately loses all his privileges to access the EHRs of Bob or any other patient. This is accomplished by simply deleting the subject membership into pseudoroles in *SPR* relation. Thus the table that represents the relation *SPR* in the BLAC model that was shown in Table 4.3, has excluded the assignment of the PCP to his pseudorole as in Table 4.6.

When Bob is assigned to a new PCP, the new PCP can immediately access all

Bob's EHRs by making simple modification to the value of Bob's attribute, *Provider.* The change is from the ID of the old PCP to the ID of the new PCP. No modifications to existing access policies or rules are needed to grant the obligatory access.

4.5 Chapter Summary

RBAC provides simple administration of access privileges, user permissions and permission review, but demands complex initial role engineering and supports only predefined and static policies. ABAC, on the other hand, simplifies initial setup, introduces flexibility and fine-granularity but increases the complexity of managing privileges and user permission review. These limitations have motivated research into the development of newer access control models that use attributes and policies while preserving RBAC's strengths, that is the simplicity of privileges and permissions administration, revocation and user permission review. Several work have been proposed in this area and their limitations have been discussed intensively in Chapter 2.

The objective of this chapter is to addresses the problem of advancing access control models and mechanisms to support the use of attributes and access control policies while preserving the advantages of RBAC. As a result, the BLAC model has been proposed. In this model, subject attributes are classified into static and dynamic based on the frequency of changes made to the attributes' values. Static attributes are used to compose *pseudoroles* that are associated with subjects, and static and dynamic attributes are used to constrain *pseudoroles* through the use of BLAC policies that are associated with objects.

The BLAC model provides bi-layer access control: when an access request is issued, the *pseudorole* is checked (first layer), and if the requester has the right *pseudorole*, the rules within the BLAC policy will be checked further for fine-grained

constraints (second layer). Although the BLAC model has been validated in the healthcare context, BLAC is generic and can be used in other domains where subjects are identified by the use of attributes as in financial services, education and critical infrastructure protection systems. Systems that require flexible, fine-grained and effective access control can utilize the BLAC model. The work by Ashutosh [9] demonstrates how BLAC model and BLAC policies can be developed for the financial domain.

This chapter has described and defined the BLAC model, discussed the generation methods of *pseudoroles*, presented policy specification and evaluation in the BLAC model, and informally validated the BLAC model via the use of use cases in the healthcare domain.

Chapter 5

An Evaluation Framework for Access Control

The literature survey in Chapter 2 discussed several access control approaches that have been proposed. However, no well-accepted framework has been designed or developed to compare various access control models, as was observed by NIST [26].

One exception to this dearth of literature was an attempt to compare access control models in terms of their expressive power by Tripunitara [90]. Tripunitara's approach measures the *ability* of an access control model to represent different access policies. That is, if all policies that can be represented in model A can be represented in model B, then model B is as expressive as model A. However, if there a policy that can be represented in A but not B, then model A is more expressive than model B.

Intuitively, in the context of this dissertation, applying a similar approach to compare the performance of different access control models would be appealing as it would permit performance comparison of the BLAC model with the traditional RBAC and ABAC models. However, this intuitive approach is not useful because

what works for expressiveness does not work for performance. Due to its use of *pseudoroles* and policies for making access decisions, the BLAC model is inherently as expressive as RBAC and ABAC.

A second exception was a more recent attempt by NIST [28] to identify evaluation metrics to measure the effectiveness of access control models based on features such as administration, enforcement, performance, and support properties [28]. Although the framework is comprehensive, the metrics used are qualitative.

Given this lack of work in this space, a new framework is needed to analyze and compare the performance of access control models; this chapter proposes and defines such an evaluation framework.

5.1 Overview of the Evaluation Framework

This section presents the evaluation framework for analyzing and comparing access control models. This framework supports quantitative analysis based on the identification of core functions of access control, which is therefore likely to lead to meaningful, numerical results.

This dissertation identifies the following functions to be core to access control models and they are defined as follows and summarized in Table 5.1.

1. *Authorization Decision-Making:* the process of evaluating access requests by access control models to grant or deny requests by subjects to perform operations on objects.

2. *Policy Modification:* the process of altering access policies by access control models to meet the needs of the organization deploying the models.

3. *Permission Modification:* the process of changing the assigning permissions of subjects to access objects by access control models.

Table 5.1: The core access control functions.

Function	**Description**
Authorization Decision-Making	Granting or denying access requests
Policy Modification	Altering access policies
Permission Modification	Adjusting rights of subjects to access objects
Revocation	Withdrawing rights of subjects to access objects
Permission Review	Determining the set of available permissions for a particular subject

4. *Revocation:* the process of deleting all assigning permissions of subjects to access objects by access control models.

5. *Permission Review.* The process of determining the set of available permissions for a particular subject.

Access control models are theoretically analyzed below on the basis of algorithmic efficiency of these five identified functions with respect to time complexity. The results of these analyses are then used to compare the models.

5.2 Applying the Evaluation Framework

This section applies the framework to RBAC, ABAC and BLAC. The specifications of the identified functions are based on the NIST model for RBAC [23] and the XACML v3.0 OASIS Standard [99] for ABAC. The specifications of the identified functions in RBAC, ABAC, and BLAC are described formally using Z notation [83], which is a specification language that formally describes and models the behavior of computing systems based on standard mathematical notation used in set theory

and first-order predicate logic. The three access control models, RBAC, ABAC, and BLAC are analyzed next.

5.2.1 Analysis of RBAC

1. *Authorization Decision-Making:* RBAC makes authorization decisions in advance as it associates permissions to roles. Thus, when a subject requests an operation over an object, RBAC checks what role is assigned with this subject, and if the requested permission (the operation over the object) is associated with the subject's role, access is granted; otherwise, access is denied.

   The **EvalAccess** function in RBAC takes as input the requesting subject s, the requested operation op and the requested object ob, and it returns a Boolean value presenting the access control decision. The pre-condition for the success of **EvalAccess** is that s is a member of S dataset, op is a member of OP dataset, and ob is a member of OB dataset. The access request by s to perform op over ob is permitted if $perm$ such that $perm \in PERM \mid PERM = \mathbb{P}(OP \times OB)$ is assigned to at least one of roles of s. The active roles assigned to s are checked to find $perm$. The set of active roles assigned to s is denoted by aR, such that $aR \in \{r : R \wedge r \in subject\_roles(s)\}$.

   In the worst case, it is observed that $|aR| \leq |\mathrm{R}|$, so **EvalAccess** is $\mathcal{O}(|R|)$, however, in practice $|aR|$ will be a small set. Consequently, the realistic complexity would be $\mathcal{O}(|aR|)$ to make an authorization decision in RBAC.

   The following schema formally describes the **EvalAccess** function.

EvalAccess

$s : NAME$
$op : NAME$
$ob : NAME$
$result! : BOOLEAN$

$s \in S;\ op \in OP;\ ob \in OB$
$result! = (\exists r : R \bullet r \in subject\_roles(s)$
$\wedge\ ((op, obj) \mapsto r) \in PA)$

2. *Policy Modification:* RBAC handles changes in policies by updating the sets of permissions associated with roles in the *PA* relation without changing permissions for each subject.

   The **ModifyPolicy** function in RBAC takes as input $op1$ and $ob1$, $op2$ and $ob2$, and r. The **ModifyPolicy** function is valid only if the pair $(op1, ob1)$ represents $perm \in PERM$, the pair $(op2, ob2)$ represents $perm \in PERM$, r is a member of R dataset, and $(op1, ob1)$ is mapped to r in PA. The **ModifyPolicy** function revokes the pair $(op1, ob1)$ assigned to r, and then grants r the pair $(op2, ob2)$. The relation PA is updated accordingly.

   In the worst case, the **ModifyPolicy** function is assumed to go through the complete set of R in the relation PA to find $(op1, ob1) \mapsto r)$ to be updated. Thus, **ModifyPolicy** is $\mathcal{O}(|R|)$ in the relation PA.

   The following schema formally describes the **ModifyPolicy** function.

ModifyPolicy

$op1 : NAME$
$ob1 : NAME$
$op2 : NAME$
$ob2 : NAME$
$r : NAME$

$(op1, ob1) \in PERM;\ (op2, ob2) \in PERM;$
$r \in R;\ ((op1, ob1) \mapsto r) \in PA$
$PA' = PA \setminus \{(op1, ob1) \mapsto r\}$
$PA' = PA \cup \{(op2, ob2) \mapsto r\}$

3. *Permission Modification:* RBAC handles the change of subject permissions by modifying subject membership into roles in the SA relation. When subject permissions change, the subject is revoked from the current role and a new subject membership can be established based on the new access requirements.

   The **PermissionModify** function takes as input $r1$, $r2$, and s. The **PermissionModify** function is valid if $r1$ and $r2$ are member of R dataset, s is a member of S dataset, and s is mapped to $r1$ and not mapped to $r2$ in the relation SA. The **PermissionModify** function deletes the assignment of s to $r1$, and establishes a new assignment of s to $r2$. The relation SA is updated accordingly.

   In the worst case, the **ModifyPermission** function is assumed to go through the complete set of S in the relation SA to find the assignment of $(s \mapsto r1)$ to be updated. Thus, **PermissionModify** is $\mathcal{O}(|S|)$ in the relation SA.

   The following schema formally describes the **ModifyPermission** function.

ModifyPermission

$r1 : NAME$
$r2 : NAME$
$s : NAME$

$r1 \in R;\ r2 \in R;\ s \in S;$
$(s \mapsto r1) \in SA;\ (s \mapsto r2) \notin SA$
$SA' = SA \setminus \{s \mapsto r1\}$
$SA' = SA \cup \{s \mapsto r2\}$

4. *Revocation:* RBAC handles subject revocation by revoking the subject membership into roles. When a subject is revoked, the subject is de-assigned from all the assigned roles in SA relation.

   The **RevokeSubject** function takes as an input the revoked subject s. The function is successful if and only if s is a member of S datasets. The **RevokeSubject** function performs subject revocation by deleting all $s \mapsto r$ in SA.

   In the worst case, the **RevokeSubject** function is assumed to go through the complete set of S in the relation SA to find all the $(s \mapsto r1)$ assignments to be deleted. Thus, **RevokeSubject** is $\mathcal{O}(|S|)$ in the relation SA.

   The following schema formally describes the **RevokeSubject** function.

RevokeSubject

$s : NAME$

$s \in S;$
$(\forall r : R \bullet r \in subject\_roles(s) \Rightarrow$
$SA' = SA \setminus \{s \mapsto r\})$

5. *Permission Review:* RBAC simplifies reviewing the available set of permissions assigned to a single subject. Permission review can be achieved by checking

the set of permissions associated with the roles assigned to the subject.

ReviewPermission takes as an input s, and returns the set of permissions assigned to the roles of a particular subject. **ReviewPermission** is valid if s is a member of S datasets. **ReviewPermission** performs permission reviewability by finding all pairs (op, ob) that represent $perm \in PERM$ such that $s \mapsto r \in SA$ and $(op, ob) \mapsto r) \in PA$ hold. **ReviewPermission** accordingly is $\mathcal{O}(|S| \times |R|)$.

The following schema formally describes the **ReviewPermission** function.

ReviewPermission

$$s : NAME$$
$$result! : \mathbb{P}(PERM)$$

$$s \in S;$$
$$(r : R;\ op \in OP;\ ob \in OB \mid (s \mapsto r) \in SA \wedge$$
$$((op, ob) \mapsto r) \in PA)$$

5.2.2 Analysis of ABAC

1. *Authorization Decision-Making:* ABAC makes access decisions at the time of requests as it compares the attributes associated with the requester against all applicable policies. If the policies are satisfied with the subject attributes, access is granted; otherwise, access is denied. When a subject makes an access request, ADE in the ABAC model checks each policy in the policy store if it applies to the request. Finally, all applicable policies will be evaluated. The time taken to find the set of applicable policies depends on the implemented policy indexing techniques. Approaches for policy indexing are outside the scope of this work; however, description of these approaches have been discussed in the XACML v3.0 OASIS Standard [99]. Rule evaluation is based on the rule combining algorithm specified in a *rule-combining algorithm-identifier*

of the specified rule-combining algorithms. XACML defines and describes 12 rule combining algorithms [99] which are used to resolve the results of the rules within each policy. To facilitate the analysis of the BLAC model (as discussed in the next subsection) and contrast it with ABAC, the chosen rule-combining algorithm is the *permit-overrides* function. The intention behind this choice is the need to compare ABAC with BLAC, and in the BLAC model, rules within a policy of the requested object are evaluated similarly to the *permit-overrides* function in XACML.

The **EvalAccess** function in ABAC takes as input the requesting subject s, the requested operation op, the requested object ob, and returns a Boolean value presenting the access control decision. The **EvalAccess** function is valid if s is a member of S dataset, op is a member of OP dataset, and ob is a member of OB dataset. The access request by s to perform op over ob is permitted when Ru exists such that it belongs to an applicable $p \in P$ and is evaluated to true.

In the worst case, all the rules Ru in p are evaluated, and all p that are applicable are evaluated. Thus, **EvalAccess** is $\mathcal{O}(|P||Ru|)$; however, in practice $|P|$ will likely be a small set that defines the set of applicable policies.

The following schema formally describes the **EvalAccess** function.

EvalAccess

$s : NAME$
$ob : NAME$
$op : NAME$
$result! : BOOLEAN$

$s \in S;\ op \in OP;\ ob \in OB$
$result! = (\forall p \in P \bullet is\_applicable(p, s) \Rightarrow$
$(\exists Ru \bullet is\_belonged(p) \wedge evaluate(Ru)))$

2. *Policy Modification:* The process of modifying policies in ABAC is intricate and usually needs human intervention. The maximum number of policies that needs to be modified in ABAC is up to 2^n where n is the number of attributes used to compose policies. Practically, however, the real number of policies that may need to be changed is small and manageable, and the need to change the complete set of policies is likely to be infrequent.

3. *Permission Modification:* ABAC allows access control decisions to depend on attributes and policies. Thus, permissions are derived directly from the attributes. In other words, in ABAC, if a subject changes his position, only his attribute *Position* needs to capture the new change. Thus, the complexity of modifying permissions in ABAC is $\mathcal{O}(1)$.

4. *Revocation:* Subject revocation can be accomplished by de-provisioning attribute assignments to subjects. However, this is not desirable in situations when a subject's information still needed to be stored. Accordingly, the process of subject revocation in RBAC is not yet well-defined.

5. *Permission Review:* The process of reviewing permissions for a given subject in ABAC is intricate as all policies need to be reviewed. For reasons similar to those stated above, the number of policies can reach 2^n where n is the number of attributes used to compose policies.

5.2.3 Analysis of BLAC

1. *Authorization Decision-Making:* BLAC, similar to ABAC, makes access decisions at the time of requests. When a subject requests an access, BLAC first checks the pseudorole associated with the subject, and then compares it with the pseudorole included in the policy associated with the requested object, not each policy in the policy store as in ABAC. If the policy is satisfied with

the subject's pseudorole, BLAC further checks the rules within the policy. If one of the rules returns true, then access is granted; otherwise, access is denied.

The **EvalAccess** function in BLAC takes as input the requesting subject s, the requested operation op, the requested object ob, and returns a Boolean value presenting the access control decision. The **EvalAccess** function is valid if s is a member of S dataset, op is a member of OP dataset, and ob is a member of OB dataset. The access request by s to perform op over ob is permitted if PRF in p of ob is satisfied by pr of s, and at least one Ru is evaluated to true.

In the worst case, all the rules Ru in p are evaluated; thus, **EvalAccess** is $\mathcal{O}(|Ru|)$.

The following schema formally describes the **EvalAccess** function.

EvalAccess

$$
\begin{aligned}
&s : NAME \\
&ob : NAME \\
&op : NAME \\
&result! : BOOLEAN \\
\hline
&s \in S;\ op \in OP;\ ob \in OB \\
&result! = (\exists p \in P \bullet p = object\_policy(ob)) \Rightarrow \\
&(\exists pr \in PR \bullet pr = subject\_pseudorole(s) \wedge \\
&is\_satisfied(pr, PRF)) \wedge \\
&(\exists Ru \bullet evaluate(Ru)))
\end{aligned}
$$

2. *Policy Modification:* BLAC, similarly to ABAC, handles the changes in policies by updating or revoking the policies that are associated with the relevant objects by humans. The number of policies that need to be modified in BLAC may reach $\mid OB \mid$. However, in practice, the actual number of policies that may need to be changed is likely to be small and manageable.

3. *Permission Modification:* Modifying permissions in BLAC, on the other hand, is slightly different from ABAC. BLAC allows the use of pseudoroles, attributes, and policies for making access control decisions. However, it is important to consider the type of attributes used for generating pseudoroles; hence, the values of these attributes should hardly change as discussed in 4.2. Thus, the complexity of modifying permissions in BLAC depends on the attribute that needs to be changed. If the attribute is not a member of the pseudorole, the complexity is $\mathcal{O}(1)$ as the attribute's value needs to be changed to reflect the new change. If the attribute is a member of the pseudorole, the complexity is based on the ModifyPermission function as described in the following paragraphs.

   The **PermissionModify** function takes as input $pr1$, $pr2$, and s and is valid if $pr1$ and $pr2$ are member of PR dataset, s is a member of S dataset, and s is mapped to $pr1$ and not mapped to $pr2$ in the relation SPR. The **PermissionModify** function deletes the assignment of s to $pr1$, and establishes a new assignment of s to $pr2$. The relation SPR is updated accordingly.

   In the worst case, the **ModifyPermission** function is assumed to go through the complete set of S in the relation SPR to find the assignment of $(s \mapsto pr1)$ to be updated. Thus, **PermissionModify** is $\mathcal{O}(|S|)$ in the relation SPR.

   The following schema formally describes the **ModifyPermission** function.

ModifyPermission

$$
\begin{array}{l}
pr1 : NAME \\
pr2 : NAME \\
s : NAME \\
\hline
pr1 \in PR;\ pr2 \in PR;\ s \in S; \\
(s \mapsto pr1) \in SPR;\ (s \mapsto pr2) \notin SPR \\
SPR' = SPR \setminus \{s \mapsto pr1\} \\
SPR' = SPR \cup \{s \mapsto pr2\}
\end{array}
$$

4. *Revocation:* BLAC handles subject revocation by simply de-assigning the subject from the assigned pseudorole in *SPR* relation.

   The **RevokeSubject** function takes as an input the revoked subject s and is valid if and only if s is a member of S dataset. The **RevokeSubject** function perform subject revocation by deleting the $s \mapsto pr$ in *SPR*.

   In the worst case, the **RevokeSubject** function is assumed to go through the complete set of S in *SPR* to find s, and thus, **RevokeSubject** is $\mathcal{O}(|S|)$.

   The following schema formally describes the **RevokeSubject** function.

   RevokeSubject

   $s : NAME$

   $s \in S$;
   $(\exists pr : PR \bullet pr \in subject\_pseudoroles(s) \land$
   $SPR' = SPR \setminus \{s \mapsto pr\})$

5. *Permission Review:* BLAC handles the process of permission review by only reviewing the set of policies where the pseudoroles within these policies match the pseudorole of the subject.

   The **ReviewPermission** function takes as an input s, and returns the set of policies that represents the authorized privileges of s. The **ReviewPermission** function is valid if s is a member of S datasets.

   ReviewPermission evaluates the complete set of P to compare *PRF* functions in $p \in P$ with pr assigned to s. **ReviewPermission** is $\mathcal{O}(|P|)$, and in BLAC, $|P| = |OB|$, thus **ReviewPermission** is $\mathcal{O}(|OB|)$.

   The following schema formally describes the **ReviewPermission** function.

ReviewPermission

$s : NAME$
$result! : \mathbb{P}(P)$

$s \in S;$
$(\forall p \in P \mid (\exists pr \in PR \bullet$
$pr = subject\_pseudorole(s)$
$\wedge\ is\_satisfied(pr, PRF))$

5.3 Comparing Access Control Models

The evaluation framework can now be used to quantify the strengths and weaknesses of each model. This section compares the BLAC model against RBAC and ABAC separately because of the lack of component similarity between RBAC and ABAC as RBAC uses *roles* and ABAC uses *attributes*. Therefore, RBAC is compared against BLAC in Section 5.3.1, and ABAC is compared against BLAC in Section 5.3.2. Table 5.2 summaries the comparison results across the three access control models for the sake of clarity in comparing the results. The highlighted values indicate the best performance. Table 5.3 and Table 5.4 summarize the comparison results respectively.

5.3.1 RBAC vs BLAC

To compare the time complexity of access evaluation algorithms of RBAC and BLAC, it is assumed that $|aR|$ assigned to a subject in RBAC is fewer than $|Ru|$ in a BLAC's policy. However, considering the high granularity provided by BLAC when compared to RBAC, the difference in the time complexity is insignificant. On the other hand, a proper comparison of the time complexity of access evaluation algorithms of RBAC and BLAC needs to consider coarse-grained access control.

Table 5.2: Summary of the time complexity of the main access control functions for RBAC vs. BLAC and ABAC vs. BLAC. The highlighted values indicate the best performance.

RBAC vs. BLAC		**Characteristic**	**ABAC vs. BLAC**	
RBAC	**BLAC**		**ABAC**	**BLAC**
$\mathcal{O}(\|aR\|)$	$\mathcal{O}(1)$	Authorization Decision-Making	$\mathcal{O}(\|P\|\|Ru\|)$	$\mathcal{O}(\|Ru\|)$
$\mathcal{O}(\|R\|)$	$\|OB\|$	Policy Modification	$\|P^{*}\|$	$\|P^{**}\|$
$\mathcal{O}(\|S\|)$	$\mathcal{O}(1)$	Permission Modification	$\mathcal{O}(1)$	$\mathcal{O}(1)$
$\mathcal{O}(\|S\|)$	$\mathcal{O}(\|S\|)$	Revocation	*undefined*	$\mathcal{O}(\|S\|)$
$\mathcal{O}(\|S\|\|R\|)$	$\mathcal{O}(\|OB\|)$	Permission Review	$\|P^{*}\|$	$\|P^{**}\|$

$^{*}P \approx 2^{n}$ $^{**}P \approx OB$

Due to the need to evaluate a single pseudorole for each access request, the time complexity of BLAC would be $\mathcal{O}(1)$ and that of RBAC would be $\mathcal{O}(|aR|)$.

Considering the policy modification algorithms of RBAC and BLAC, RBAC performs better than BLAC. RBAC is $\mathcal{O}(|R|)$, and can be made to perform even better depending on the data structures used to store R and/or the searching algorithm used to find $r \in R$. On the other hand, BLAC needs human intervention for policy modification.

Permission modification is flexible in BLAC due to its use of attributes. Assuming changes in permissions do not affect attributes that are used to generate pseudoroles, and because permissions are derived directly from attributes, the time complexity for modifying permissions in BLAC is $\mathcal{O}(|1|)$. However, RBAC is $\mathcal{O}(|S|)$ due to its need to modify the assignment of $s \mapsto r1$ in SA relation.

Due to the use of pseudoroles in BLAC and roles in RBAC, the time complexity of subject revocation algorithms in both models is $\mathcal{O}(|S|)$ as RBAC and BLAC need

to delete the assignment of all $s \mapsto r$ in SA in RBAC and $s \mapsto pr$ in SPR in BLAC.

The process of permission review in RBAC is simpler and can be quantified as $\mathcal{O}(|S| \times |R|)$. In BLAC, and due to the use of policies for making access decisions, reviewing the set of permissions for a particular subject needs to compare a subject's pseudorole against all pseudoRoles functions in the policies associated with objects. It is quantified as $\mathcal{O}(|OB|)$. The product of the number of subjects and roles in a system is much fewer than objects. Permission review is possible in BLAC, as in RBAC, but causes additional overhead.

Table 5.3: Summary of the time complexity of the core access control functions in RBAC and BLAC.

Characteristic	**RBAC**	**BLAC**
Authorization Decision-Making	$\mathcal{O}(\|aR\|)$	$\mathcal{O}(1)$
Policy Modification	$\mathcal{O}(\|R\|)$	$\|OB\|$
Permission Modification	$\mathcal{O}(\|S\|)$	$\mathcal{O}(1)$
Revocation	$\mathcal{O}(\|S\|)$	$\mathcal{O}(\|S\|)$
Permission Review	$\mathcal{O}(\|S\|\|R\|)$	$\mathcal{O}(\|OB\|)$

5.3.2 ABAC vs. BLAC

The time complexity of the ABAC access evaluation algorithm depends on the process of finding the set of applicable policies among the complete list of policies, and the number of applicable policies and rules that need to be evaluated. Here, it is assumed that the algorithm uses policy indexing techniques. Thus, the time complexity is limited to the process of evaluating policies and rules within these policies, which is $\mathcal{O}(|P||Ru|)$.

On the other hand, BLAC evaluates a single policy that is associated to the requested object; thus, time complexity is limited to the process of evaluating the associated policy and the rules within this policy, which is $\mathcal{O}(|Ru|)$. BLAC performs well compared to ABAC as it only needs to evaluate a single policy. Assuming there are ten 1-rule policies in ABAC and a single 10-rule policy in BLAC that need to be evaluated, BLAC performs better than ABAC because ABAC needs to combine individual decisions into a single aggregated decision by policy-combining algorithms [99] which BLAC does not need to do.

In both ABAC and BLAC, modifying policies needs human intervention. Although the number of policies to modify in both model can be up to 2^n in ABAC and $\mid OB \mid$ in BLAC, there is no significant difference between the two models. Also, in practice, the number of policies that may need to be modified is usually small and manageable, and the need to update the complete set of policies is small.

Permission modification is flexible in both ABAC and BLAC due to the use of attributes, thus it is $\mathcal{O}(|1|)$ in both models. In short, there is no noteworthy difference between the two models.

Subject revocation in ABAC is not a well-defined process, which makes it difficult to quantify time complexity. On the other hand, the process is well-defined in BLAC due to the use of pseudoroles which can be quantified to $\mathcal{O}(|S|)$ to delete the assignment of $s \mapsto pr$ in SPR.

The process of permission review in BLAC has an overhead which is similar to ABAC's. In BLAC, reviewing the set of permissions for a particular subject needs to compare subject's pseudorole against all PRF functions in the policies; thus, it can be quantified as $\mathcal{O}(|OB|)$. In ABAC, all policies need to be executed to review the set of permissions of a single subject.

From the previous discussion, note that the comparison between ABAC and BLAC is primarily based on the number of polices in both models.

Table 5.4: Summary of the time complexity of the core access control functions in ABAC and BLAC.

Characteristic	**ABAC**	**BLAC**
Authorization Decision-Making	$\mathcal{O}(\|P\|\|Ru\|)$	$\mathcal{O}(\|Ru\|)$
Policy Modification	$\|P\|$	$\|P\|$
Permission Modification	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Revocation	*undefined*	$\mathcal{O}(\|S\|)$
Permission Review	$\|P\|$	$\|P\|$

A realistic assumption for a medium-sized enterprise is based on the Rochester General Hospital System (RGHS), which is a hospital system in Monroe County, New York [69]. Here it makes sense to assume there are up to 10,000 subjects (users) and 100 million objects that need to be accessed. For such an enterprise, the following three cases need to be considered for the range in the numbers of policies for ABAC: small, medium and large; the discussion below provided the numbers of policies for BLAC.

The following three sub-sections present assumptions about attributes, generate appropriate numbers of policies in ABAC and BLAC, and then compare ABAC against BLAC based on these assumptions.

Small number of ABAC policies

Assumptions. A system here may be assumed to have up to 10 attributes: four subject attributes, two object attributes, two operation attributes and two environment attributes. Under ABAC, this set of attributes could result in up to 2^{10} polices, i.e., 1,024 polices. However, under BLAC, the number of policies is 100 million

assuming that each object has a distinct policy; however, this assumption is impractical as in the healthcare domain multiple health records have the same access permissions. Polices in both models are assumed to have one rule for consistency.

Comparison between ABAC and BLAC. In ABAC, the time complexity of the access evaluation algorithm focuses on the evaluation of the applicable policies and rules within these policies. Thus, in the worst case, the time complexity is $\mathcal{O}\ (1024)$ because all the policies are considered applicable and need to be evaluated. In BLAC, the time complexity of access evaluation algorithm is limited to the process of evaluating the single distinct associated policy, thus, it is $\mathcal{O}(1)$. Noticeably, BLAC outperforms ABAC because it only needs to evaluate a single policy.

In ABAC, the number of policies that may need to be modified is 1024. On the other hand, the number of polices in BLAC is 10^8. In practice, however, the number of policies that may need to be modified is usually small and manageable, and the need to update the complete set of policies is rare. On the other hand, ABAC outperforms BLAC when it comes to modifying polices.

In ABAC, the number of policies that may need to be reviewed to find the set of permissions for a particular subject is 1024 because all policies need to be checked. In the worst case for BLAC, the time complexity is $\mathcal{O}(10^8)$ because all policies may be assumed to be satisfied with the specified subject's pseudorole. ABAC thus outperforms BLAC in the review of the set of permissions for a particular subject.

Medium Number of ABAC Policies

Assumptions. For this case, assume a system had 30 attributes: 15 subject attributes, five object attributes, five operation attributes and five environment attributes. Under ABAC, this set of attributes may result in up to 2^{30} polices, i.e., $1.073\mathrm{e}^9$ polices. Under BLAC, the number of policies is 100 million based on the

same assumptions as in the first case.

Comparison between ABAC and BLAC. In ABAC, the time complexity of access evaluation algorithm is $\mathcal{O}(2^{30})$. In BLAC, the time complexity of access evaluation algorithm is again $\mathcal{O}(1)$. BLAC outperforms ABAC due to the need to evaluate a single policy.

The number of policies that may need to be modified is (2^{30}) in ABAC and (10^8) in BLAC. Again, BLAC outperforms ABAC in modifying polices.

The number of policies that may need to be reviewed to find the set of permissions for a particular subject is (2^{30}) in ABAC and (10^8) in BLAC. Thus, BLAC notably outperforms ABAC in reviewing the set of permissions for a particular subject.

Large Number of ABAC Policies

Assumptions. Here consider a system with 100 attributes: 75 subject attributes, 10 object attributes, five operation attributes and 10 environment attributes. Under ABAC, this set of attributes could result in 2^{100} polices, i.e. $1.267\mathrm{e}^{30}$ polices. Under BLAC, the number of policies is 100 million based on the same assumptions as in the first two cases.

Comparison between ABAC and BLAC. In ABAC, the time complexity of access evaluation algorithm is $\mathcal{O}(2^{100})$. In BLAC, the time complexity of access evaluation algorithm is again $\mathcal{O}(1)$. BLAC significantly outperforms ABAC because it only needs to evaluate a single policy.

The number of policies that may need to be modified is (2^{100}) in ABAC and (10^8) in BLAC. Again, BLAC significantly outperforms ABAC in modifying the set of polices.

The number of policies that may need to be reviewed to find the set of permissions for a particular subject is (2^{100}) in ABAC and (10^8) in BLAC. Thus, BLAC significantly outperforms ABAC in reviewing the set of permissions for a particular subject.

5.4 Chapter Summary

This chapter examined the performance of the BLAC model. For this purpose, an evaluation framework was proposed to theoretically evaluate and compare access control models using five core functions of access control. The framework was used to evaluate RBAC, ABAC, and BLAC based on time complexity. The results of these evaluations were then used to compare the BLAC model against RBAC and BLAC.

As detailed in Section 5.3, the BLAC model outperforms RBAC in terms of authorization decision-making and permission modification, and performs comparably to RBAC in revocation. Due to the use of policies in BLAC, however, RBAC outperforms BLAC in policy modification and user permission review. On the other hand, the BLAC model outperforms ABAC in terms of authorization decision-making and revocation, and performs comparably to ABAC in policy and permission modification and user permission review.

In this chapter, the focus was a theoretical analysis and comparison for the efficiency of various algorithms underlying access control models. Another possible approach is to model and implement/simulate systems based on the model, and subsequently compare the implementation/simulation based on the running time of the identified functions or algorithms. For example, the running time to evaluate access requests or the time to update permissions of subjects. This approach, however, is difficult to justify because the best way to implement such models is

not known and thus the results need to be proved [90]. Moreover, to make a realistic empirical comparison, all models have to be implemented or simulated using identical enterprise-level parameters, which is beyond the scope of this dissertation. A theoretical comparison based on time complexity, as shown here, remains the most appropriate approach for this dissertation.

Chapter 6

An Insider Threat Model for Access Control

As discussed earlier, information sharing has become crucial in modern computing applications in a variety of domains. For example, in the healthcare domain, the United States Health Information Technology for Economic and Clinical Health Act (HITECH) of 2009 encourages healthcare providers to share information with one another, as well as patients, for improving healthcare quality and lowering costs [93]. These benefits of sharing information need to be balanced with security and privacy concerns, especially when personally identifiable information is involved. HIPAA and GLBA specify strict requirements for the protection of information [21, 94]. A major requirement specified by HIPAA and GLBA is access control.

The increase in reported incidents of successful insider attacks shows that currently deployed access control mechanisms are inadequate in protecting against such attacks. The Privacy Rights Clearinghouse tracks data breaches [65] that typically have compromised data elements such as social security numbers, account numbers, and driver's license numbers that can be exploited by rogue insiders.

These reported breaches are categorized by organization types such as healthcare, educational institutions, government and military, businesses (retail, financial, and other) and non-profits [65].

The healthcare domain is particularly sensitive and it receives more than its share of attacks, especially from insiders. In a PricewaterhouseCoopers survey of more than 600 healthcare providers, insurers, pharmaceuticals, and life sciences professionals, 40% reported an improper use of protected health information by internal parties [66]. A recent report [67] stated that U.S. and German companies experience the most expensive data breach incidents while Brazil and India had the least costly data breaches. Several other reported healthcare information breaches by insiders [20, 58, 70, 98] have cost healthcare organizations in penalties between $50,000 for one-time violations to $1.5 million for repeat violations across all HIPAA violation categories [44].

The ease and frequency of such inappropriate accesses compels an examination of traditional and current approaches used for access control in healthcare systems, and particularly how these approaches handle threats and attacks, especially from *insiders*. The impact of insider attacks can be significantly worse than that of outsider attacks [3, 12]. One major reason is that insiders already have authorized credentials that allow them some level of access within an organization, thus leading to easier opportunities to cause damage. One approach of assessing how current access control mechanisms mitigate insider threats is evaluating these methods against an insider threat model.

Given the lack of a formal threat model designed specifically for access control, this chapter introduces an insider threat model based on a holistic approach

toward modeling access control in systems. Threat modeling is a process for understanding and analyzing the security of a system by following a systematic approach to identify potential security threats to the system [30, 85]. Given the sensitivity of healthcare data, this model is constructed in the context of a healthcare application, but the threat model itself is sufficiently general and can be applied to access control as used in other domains. The chapter also provides an assessment of how the constructed insider threat model performs against the two major access control models, Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC), as well as a new access control model, BiLayer Access Control model (BLAC) [7], that was proposed to combine the best features of RBAC and ABAC.

The rest of this chapter summarizes the background in threat model methodologies, designs and constructs an insider threat model for access control in healthcare systems, and uses the constructed threat model to assess the effectiveness of access control approaches for mitigating insider threats.

6.1 Threat Modeling Methodologies

This section reviews the related methodologies in the threat modeling area. Threat modeling helps to address top threats that may have the biggest potential to impact a system. Several threat modeling methodologies exist to construct threat models, including Microsoft Threat Modeling Methodology [50], Microsoft Threat Modeling for Web Applications [51], OWASP Application Threat Modeling [88], Process for Attack Simulation and Threat Analysis (PASTA) [45], and Trike [60].

Microsoft Threat Modeling Methodology [50] and OWASP Application Threat Modeling [88] start with identifying assets that could attract attackers, understanding the target application by creating use-cases to understand how the application

could be used, identifying entry points to determine how attackers could interact with the application, and analyzing data flow diagrams (DFDs) to demonstrate how data travels through the different paths in the application. Next, potential threats are identified using a threat categorization methodology such as Microsoft STRIDE model [52], or the Application Security Frame (ASF) [53].

PASTA [45], and Trike [60] differ from Microsoft and OWASP threat modeling. The former identifies business objectives and security and compliance requirements, and the latter takes risks into perspective. In this work, the Microsoft Threat Modeling Methodology [50] was adopted for constructing the insider threat model, because it was the most suitable method for access control mechanisms.

6.2 Threat Model Construction

This section presents the threat model for the generic access control system. The main objective for constructing the threat model in this chapter is to help improve the security of applications from the perspective of access control. To explain the threat model construction process, a healthcare application is used. However, the approach used to construct this threat model is sufficiently general.

The focus of concern in this threat model being constructed here is the protection of *patient healthcare data* from unauthorized or improper use and disclosure (*confidentiality*), and unauthorized or improper modification and destruction (*integrity*) by healthcare providers. Patient healthcare data, as used here, primarily refers to electronic protected health information (e-PHI), as described in the HIPAA Security Rule [92], that is created, received, maintained or transmitted in an electronic form by healthcare providers.

The *use* of healthcare data is defined by HIPAA as the sharing, employment,

application, utilization, examination, or analysis of protected healthcare information within organizations that maintain such information [95]. HIPAA specifically defines the *disclosure* of healthcare data as "the release, transfer, provision of, access to, or divulging in any other manner of information outside the entity holding the information" [95]. The ability to carry out operations over e-PHI, including the use, disclosure, modification and destruction, is denoted as *access* [96].

Access to healthcare data is classified as authorized and unauthorized based on a set of access policies defined by healthcare organizations or healthcare laws and regulations. Authorized access in turn can be, however, classified as either proper or improper. More formal definitions of these terms are provided below, and Figure 6.1 illustrates the relationship among these access types.

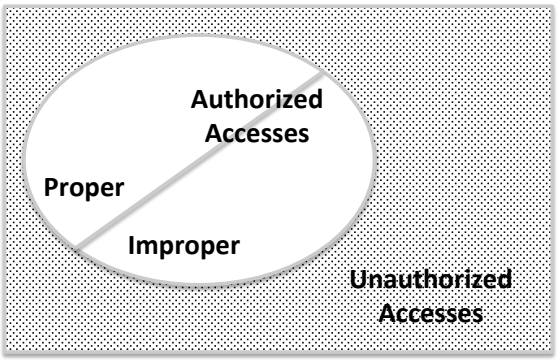

Figure 6.1: The classification of access in healthcare applications showing the two main type of access: authorized access (white area) unauthorized access (patterned area) based on access policies. Authorized access are further classified to proper and improper access.

- **Authorized access:** healthcare providers have access rights to healthcare data according to the set of security policies enforced by a healthcare application.
- **Unauthorized access:** healthcare providers have no access rights to the data, but have deliberately circumvented the application to gain access.

- **Improper access:** healthcare providers have access rights to the data granted to them by the application, but have used their access to perform operations they are not truly entitled to.

As the focus is on insider attackers, the main adversaries are the authorized users, i.e., insiders, who have some level of authority to access data depending on their identity attributes, and the security policies specified by their healthcare organization.

The following paragraphs present and describe the steps to generate the threat model.

1. Identifying the security objective

This step permits the model builder to focus on the process of constructing the threat model. The main security goal of this threat model is to minimize unauthorized and improper use, disclosure, modification, and destruction of patient healthcare data by insiders, based on a set of access policies defined by healthcare organizations and healthcare laws and regulations.

2. Creating the application overview

This step permits the model builder to understand the main functionalities and subjects of the target application. Identified here are the application architecture including the application key components, main usage scenarios, roles of subjects, and how the application components interact with each other and with external entities, i.e., healthcare providers. Based on the purpose of the threat model, the identification of these items is tied to the access control.

The overall architecture of a general healthcare application implementing a generic access control model is illustrated in Figure 6.2. To understand the target healthcare application fully, a *use case* from the healthcare domain is defined to describe the main usage scenarios and roles of subjects.

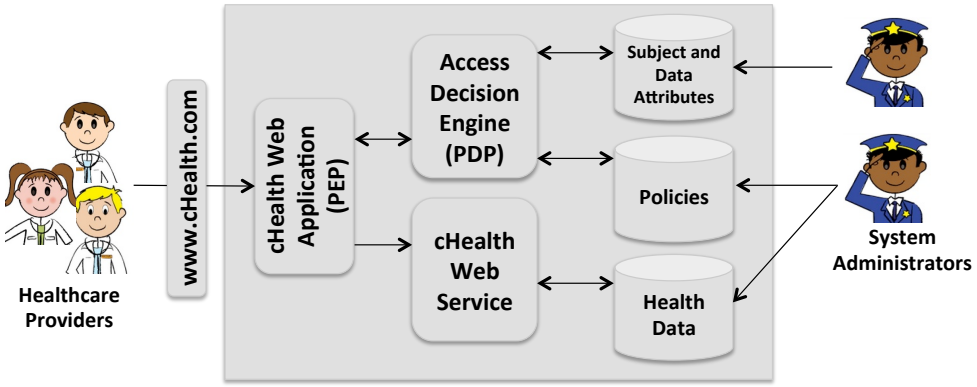

Figure 6.2: A general architecture of a healthcare application implementing a generic access control model.

In a medical center with two hospital affiliates, hospital A and hospital B, multiple healthcare providers use a healthcare application called "cHealth" to manage patients' healthcare data in both hospitals. The roles of healthcare providers can be physicians, nurses, and administrative and billing staff, in addition to application administrators to maintain the application and access polices. Each healthcare provider is defined by a set of attributes, for example *name*, identification number (*ID*), *gender*, the field of the healthcare *provider*, their *department*, and their office *location*. These attributes are stored in a database.

Healthcare data is stored in another database in a file-based form that conforms to XML specifications. Healthcare data is defined by attributes, for example, patient name, patient MRN (Medical Record Number), patient DOB (Date of Birth), and the ID of the physician responsible for treating the patient. These attributes are also stored in a database. User and data attributes are typically provided and managed by trusted entities, however, managing attributes is out of scope for this dissertation. Healthcare data is organized into a hierarchical data structure: (1) demographical, (2) clinical, and (3) billing, to provide fine-grained access control.

Typical usage scenarios are identified below to describe cHealth characteristics.

- Physicians and nurses create, read, and modify the demographical and clinical sections for patients who are under their responsibility in normal situations, with the exception of psychotherapy notes.
- Physicians and nurses create, read, and modify the demographical and clinical sections for non-patients in emergency situations, with the exception of psychotherapy notes.
- Healthcare providers do not delete data in any section.
- Administrative staff create, read, and modify data within the demographical section when they are on duty.
- Billing staff create, read, and modify data within the billing section and read data within the demographical section when they are on duty.
- Application administrators delete data after a predefined time.
- Healthcare providers generate access policies for the newly created data.
- Application administrators modify access policies for the created data.

Healthcare providers are entitled to access patient healthcare data through their web browsers based on their identity attributes, and according to their organizations' policies. The interactions among components of the cHealth application and healthcare providers are controlled by the access control model to grant or deny access requests. These interactions are described in Figure 6.3, and listed below.

1. Subject logs in and requests data through cHealth Web Application.
2. cHealth Web Application creates a web request and sends it to Access Decision Engine.

3. Access Decision Engine retrieves relevant policies.

4. Access Decision Engine retrieves attribute values related to subject, data, or environment.

5. Access Decision Engine makes access decision based on the access control model, and sends access decision to cHealth Web Application.

6. cHealth Web Application enforces authorization decision: if accept, cHealth Web Application permits subject to access and perform requested operation over requested data via cHealth Web Service (6a). If deny, cHealth Web Application rejects subject's access request (6b).

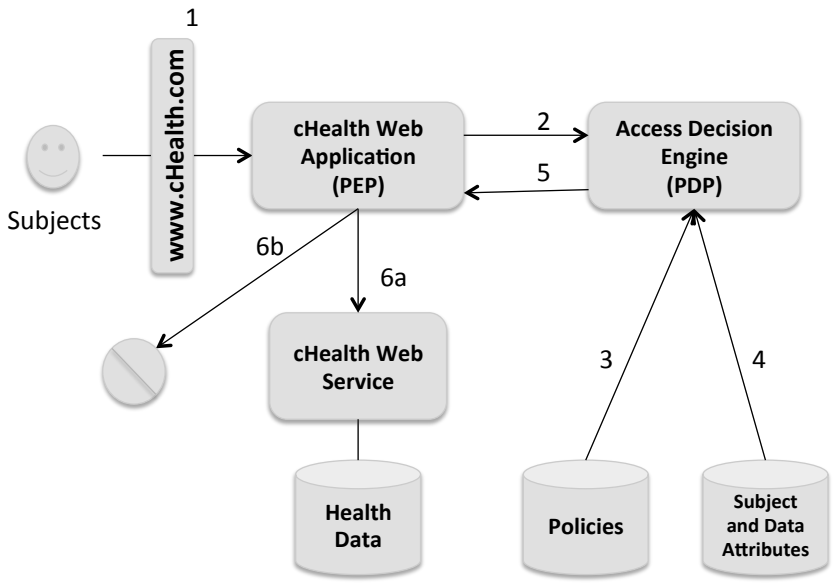

Figure 6.3: The interactions among the components of access models start when a subject logs in and requests data through the application, then the application sends the request to the ADE which retrieves relevant policies and attributes. The ADE makes an access-control decision and sends the decision to the application for enforcement: if accept, subject is forwarded to the web service.

3. Decomposing the healthcare application

This step helps the model builder understand the target application in detail and how the internal components interact with one another and with external entities. The data flows and entry and exit points within the healthcare application are identified.

Figure 6.4 shows a high-level data flow diagram (DFD) between the application components. The purpose of the DFD is to understand how data is processed within the internal components. The rectangles denote external entities, and circles represent functions performed on data, or performed on other functions based on data. The two parallel lines and curved and directional arrows indicate databases and data movement. The curved and dashed arrows represent trust boundaries that refer to changes in access control levels as data flows through the application.

Entry and exit points refer to the interfaces that external entities use to interact with the application to send requests, process data, respond to requests, or send data. In the healthcare application, the page that subjects use to log in to the cHealth Application before requesting data access is considered an entry point. It is denoted as the first step in the interaction process based on the access control model illustrated in Figure 6.3.

The cHealth main page is an entry and exit point for all successfully logged-in subjects to carry out one or more of the usage scenarios identified earlier. As the goal of the desired threat model is to identify threats posed by insiders, the cHealth main page is the only point considered as it is controlled by the access control model in order for the subjects to perform operations over data.

4. Identifying the threats

This step permits the model builder to identify relevant threats that may compromise the security objective. Generating an attack tree is a method of representing threats against an application in a graphical or outline form [76]. An attack tree

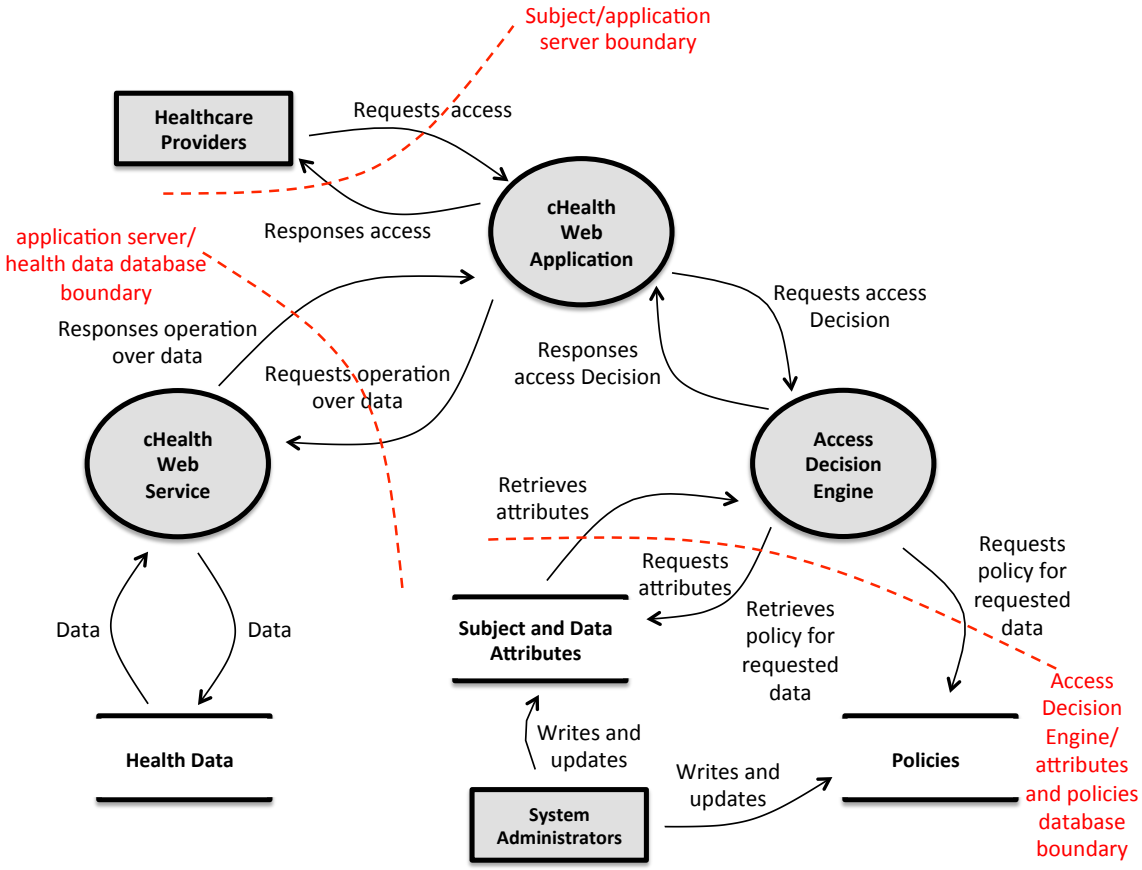

Figure 6.4: The data flow diagram showing how the data processed within the internal components. Rectangles, circles, two parallel lines, curved and directional arrows, and curved and dashed arrows denote external entities, functions, databases, data movement, and trust boundaries respectively.

consists of a root node and child nodes, where the root node denotes a threat, and child nodes represent various methods to realize that threat. An outline for the attack tree for the healthcare application security objective is shown in Figure 6.5.

The construction of the threat model results in the effective identification of a set of threats and alternative approaches used to launch these threats that are relevant to the security objective. In the next section, access control models are briefly assessed against the identified threats to test their efficacy in mitigating the risk of insider threats.

```
Threat 1: Unauthorized access of health information (use, disclosure, alteration,
and destruction) by healthcare providers
1.1: Gain authorized healthcare provider's credentials
1.1.1: Ask for authorized healthcare provider's credentials
1.1.1.1: Ask for a temporary use of password
1.1.1.2: Corporate with an authorized healthcare provider
1.1.1.3: Fool an authorized healthcare provider to leak credentials
1.1.2: Steal authorized healthcare provider's credentials
1.1.2.1: Phishing
1.1.2.1.1: Email
1.1.2.1.2: Fake website
1.1.2.2: Implant malware
1.1.2.3: Install keystroke hardware
1.1.2.4: Shoulder surfing
1.2: Obtain access credentials
1.2.1: Brute Force
1.2.2: Use default credentials
1.2.3: SQL injection
1.2.4: Monitor network traffic
1.3: Use unattended logged-in machine
1.4: Steal authorized healthcare provider's machine
Threat 2: Improper access of health information (use, disclosure, alteration, and
destruction) by healthcare providers
2.1: Use their own credentials
```

Figure 6.5: The generated attack tree showing two identified threats: unauthorized access of data with four various methods to launch it, and improper access of data with one method to launch it.

6.3 Access Control Models Assessment

This section evaluates how the three access control approaches, RBAC, ABAC and BLAC, perform against the identified threats by insiders (i.e., healthcare providers themselves). Two fundamental types of threats exist: (1) unauthorized access of information, and (2) improper access of information. Access again refers to the set of operations—the use, disclosure, alteration, and destruction of data—that healthcare providers may perform without authorization or improperly over healthcare information.

6.3.1 Evaluating Access Control Models Against Unauthorized Access Threat

Healthcare providers, or attackers, can gain unauthorized access to healthcare information via several methods. Note that the same methods can also be carried out by outsiders, i.e., unauthorized users who have no access to data but try to gain such access by illegitimate means; however, the focus here is on attacks launched by insiders.

Insiders may obtain credentials from legitimate healthcare providers who are authorized to access the target healthcare information in several ways including:

- Asking for and obtaining credentials from authorized users,
- Using authorized users' unattended logged-in machines, or
- Stealing or illegally obtaining credentials from authorized users,
- Stealing devices that contain the credentials of authorized users, and
- Stealing devices or storage that contains the target protected heath information.

In these cases, the insiders are able to break the *authentication* scheme being used. That is, the application maps these insiders (attackers) to the identity attributes associated with the authorized healthcare providers. If the attributes associated with authorized providers, along with the attributes associated with the object, action, and environment, satisfy the policy of the target healthcare information being attacked, the attackers (insiders) would be able to access the target information.

In other words, the strength of the access control model to guard against unauthorized access depends on the robustness of the authentication scheme being used. Due to their utilization of attributes, ABAC and BLAC will be able to prevent attacks as the attacker can only spoof subjects, but not attributes. In RBAC,

however, the insider attacker is likely to have access to a larger subset of healthcare information due to RBAC's lack of granularity. Both ABAC and BLAC utilize attributes and policies, all of which must be satisfied to grant access, thus reducing the subset of healthcare information that can be threatened by the attacker.

6.3.2 Evaluating Access Control Models Against Improper Access Threat

Authorized healthcare providers may be able to perform improper operations over healthcare information using their own credentials. Such improper access may be possible as most healthcare applications that implement RBAC are typically regulated using *the role of healthcare providers*. That is, once a set of healthcare providers are assigned to a role, all providers assigned to this role will be assigned to the same permission set.

Such an assignment does not take into account the providers' involvement in the treatment of each patient, as required by the HIPAA Privacy Rule [91]. In such an RBAC setting, it is possible for healthcare providers to gain improper access. Even the use of auditing mechanisms that may log such improper access is not sufficient as improper access would have already occurred; the goal here must be the prevention, not the subsequent detection.

Due to the fine granularity and high flexibility of ABAC and BLAC, the set of healthcare information that providers can access is further constrained by various attributes and a set of fine-grained access control policies. For example, when using the BLAC model, it is feasible to specify that healthcare providers can only access health information of patients that these providers have direct treatment relationships with. Also, access by emergency department providers can be limited to access requests within the hospital locations. Thus, the BLAC model would significantly decrease the possibility of improper actions and the set of exposed

healthcare information comparing to RBAC, due to the use of attributes and fine-grained access policies.

In summary, the analysis shows the fine-grained features of both ABAC and BLAC enable them to mitigate insider threats better than RBAC. Compared to ABAC, BLAC has reduced complexity of access control evaluation, user revocation and user permission review as discussed in Chapter 5. Among these three schemes, BLAC is shown to be the most effective access control approach for mitigating insider threats.

6.4 Chapter Summary

Access control mechanisms can often mitigate unauthorized access by external subjects, i.e., outsiders, but it is more challenging to mitigate insider threats as they already have some authority to access data in the system. This chapter designed and constructed a threat model for access control to address the security objective of minimizing unauthorized and improper use, disclosure, modification, and destruction of patient health information by insiders.

The constructed model was used to identify two insider threats: unauthorized access and improper access of health information by healthcare providers. Although a healthcare application was used for demonstration, the threat model constructed here and the identified threat can be applied to other domain areas.

The threat model constructed in this chapter was then used to evaluate the effectiveness of current access control models, RBAC, ABAC and BLAC. The analysis indicated that ABAC and BLAC mitigate insider threats better than RBAC. However BLAC has lower complexity than ABAC, and thus is likely to be more efficient.

Chapter 7

Summary

This chapter concludes this dissertation and discusses several possible future research directions.

7.1 Conclusion

This dissertation investigated the use of attributes and policies in the design of effective access control models to address the problem of improving access control models while maintaining the advantages of RBAC.

The dissertation presented an access control design using CP-ABE that provided effective solutions to some of the issues related to standard access control mechanisms. It also explored the viability of adopting CP-ABE in terms of time and storage overhead. The results suggested that the proposed design would provide promising performance and consume trivial storage, and thus CP-ABE can be used to enforce access control.

The dissertation also proposed the BLAC model, which is a central contribution of this work. The BLAC model uses pseudoroles and BLAC polices to perform a two-step access control evaluation. The first step checks access requests to verify whether requesting subjects have the pseudoroles specified in the BLAC policies of

the requested objects. If requesting subjects hold the right pseudoroles, the second step checks rule(s) within the associated BLAC policies for additional constraints on access.

The dissertation then defined an evaluation framework to examine the performance of the BLAC model. The framework consists of five core functions that are used to evaluate access control models based on the time complexity. The results of these evaluations are then used to compare the BLAC model against RBAC and ABAC.

BLAC outperforms RBAC in terms of authorization decision-making and permission modification, and performs comparably to RBAC in revocation. Due to the use of policies in BLAC, however, RBAC outperforms BLAC in policy modification and permission review. On the other hand, the BLAC model outperforms ABAC in terms of authorization decision-making and revocation, and performs comparably to ABAC in policy and permission modification and permission review.

The dissertation finally constructed a generic access-control threat model to analyze the effectiveness of the BLAC model in mitigating insider threats and compare it against RBAC and ABAC. The analysis shows that BLAC is able to decrease unauthorized and improper access, with better performance.

7.2 Future Work

This dissertation provides an extensive investigation of the use of attributes and policies in access control models. As a result of this dissertation, several useful areas of research have been identified for further investigation, as described next.

A Large Scale Empirical Study of the BLAC Model

Although this dissertation validates the practical viability of BLAC by applying it to the healthcare domain in Section 4.4 and evaluates its performance through a

theoretical analysis with respect to the time complexity, an additional large-scale empirical study can be further conducted.

A possible approach for the large-scale empirical study is to model access control models implemented in production systems, including RBAC, ABAC, and BLAC, and then implement these models. Consequently, simulation scenarios using the models can be run and analyzed in terms of the functions identified in Table 5.1.

Access Control In Big Data Systems

Another interesting area that builds on the work done in this dissertation is the application of access control models to Big Data systems. In the past few years, such systems including Apache Hadoop [87] and Cassandra [86] have become critical in a world of exponentially increasing information. In order to comply with various privacy acts and regulations such as HIPAA [94], access to such information needs to be controlled.

These big data systems, however, do not support access control, whether it is coarse-grained or fine-grained. These systems thus suffer from all of the limitations discussed throughout this dissertation. An investigation of access control issues using RBAC, ABAC, and especially BLAC, in big data systems seems worthwhile.

Logical Evaluation Framework for Access Control

This dissertation discussed the need for a framework to compare various access control models. Due to the lack of work in this space, this dissertation presented an evaluation framework that allows comparing access control models based on the algorithmic efficiency considering the time complexity. This work leads to another interesting direction in the area of evaluation and comparing access control

models.

Access control models have more elements beyond efficiency that need to be captured. These elements may include the level of granularity and the ease of development. Thus, a logical evaluation framework for assessing these factors is worth exploring.

The Integration of the BLAC Model and CP-ABE

The BLAC model is designed to use attributes and policies for making access decisions in a linear manner with respect to the number of attributes used. CP-ABE, on the other hand, uses attributes and policies to encrypt data and associates encrypted data with policies defining who can decrypt it.

A framework for integrating or combining BLAC and CP-ABE is worth investigating, because the integrated approach is likely to result in more secure and more flexible systems via combining the access control and encryption mechanisms. The integrated framework is likely to perform comparably to BLAC and CP-ABE or better.

Privacy-Preserving in the BLAC Model

A critical issue related to this dissertation is privacy preservation in attribute-based access control models in general and in BLAC specifically. This issue is motivated by the fact that various kind of attributes used in making access requests and policies may reveal private information [55,78]. In the BLAC model, attributes of a requested object and requester are sent to the Access Decision Engine, usually hosted in a remote server provider, for evaluation. Access policies are also stored in a remote server provider. Thus, the BLAC model is vulnerable to such threats.

Attributes and access policies must be protected to ensure preserving two main privacy properties: requester anonymity and relationship anonymity [80]. In the

BLAC model, requester anonymity guarantees the difficulty of determining the identities of access requesters. Relationship anonymity, on the other hand, ensures the separation of access requesters and requested objects.

Bibliography

[1] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin. Charm: a framework for rapidly prototyping cryptosystems. *Journal of Cryptographic Engineering*, 3(2):111–128, 2013.

[2] M. A. Al-Kahtani and R. Sandhu. A model for attribute-based user-role assignment. In *Proceedings of the 18th Annual Computer Security Applications Conference*, ACSAC '02, Washington, DC, 2002. IEEE Computer Society.

[3] M. Alawneh and I. M. Abbadi. Defining and analyzing insiders and their threats in organizations. In *Proceedings of the 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, TRUSTCOM '11, pages 785–794, Washington, DC, USA, 2011. IEEE Computer Society.

[4] H. Alipour, M. Sabbari, and E. Nazemi. A policy based access control model for web services. In *Internet Technology and Secured Transactions (ICITST), 2011 International Conference for*, pages 472 –477, Dec 2011.

[5] S. Alshehri, S. Mishra, and R. K. Raj. Insider threat mitigation and access control in healthcare systems. Technical report, Rochester Institute of Technology, Department of Computer Science, May 2013. Available at: `https://ritdml.rit.edu/handle/1850/16660`.

[6] S. Alshehri, S. P. Radziszowski, and R. K. Raj. Secure Access for Healthcare Data in the Cloud Using Ciphertext-Policy Attribute-Based Encryption. In *ICDE Workshop on Data Management in the Cloud, DMC '12*. IEEE, 2012.

[7] S. Alshehri and R. K. Raj. Secure Access Control for Health Information Sharing Systems. In *IEEE International Conference on Healthcare Informatics, ICHI 2013*. IEEE, 2013.

[8] Amazon. Amazon Simple Queue Service - Basic Use Cases for Access Control, November 2012. `http://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/AccessPolicyLanguage_UseCases.html`.

[9] F. Ashutosh. Effectiveness of Bi-Layer Access Model in the Financial Domain. Master's thesis, Rochester Institute of Technology, 2014.

[10] J. Benaloh, M. Chase, E. Horvitz, and K. Lauter. Patient controlled encryption: ensuring privacy of electronic medical records. In *Proc. 2009 ACM Workshop on Cloud Computing Security*, 2009.

[11] E. Bertino, P. A. Bonatti, and E. Ferrari. Trbac: a temporal role-based access control model. In *Proceedings of the fifth ACM workshop on Role-based access control*, RBAC '00, pages 21–30, New York, NY, USA, 2000. ACM.

[12] E. Bertino and G. Ghinita. Towards mechanisms for detection and prevention of data exfiltration by insiders: keynote talk paper. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '11, pages 10–19, New York, NY, USA, 2011. ACM.

[13] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *Proc. 2007 IEEE Symposium on Security and Privacy*, 2007.

[14] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized trust management. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, SP '96, Washington, DC, 1996. IEEE Computer Society.

[15] P. Bonatti and P. Samarati. Regulating service access and information release on the web. In *Proceedings of the 7th ACM conference on Computer and communications security*, CCS '00, pages 134–143, New York, NY, 2000. ACM.

[16] D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. In *Proc. 21st Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '01. Springer, 2001.

[17] CareCloud. Carecloud, web-based medical practice management software, 2011. `http://www.carecloud.com/`.

[18] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina. Controlling data in the cloud: outsourcing computation without outsourcing control. In *CCSW '09, Workshop on Cloud Computing Security*, 2009.

[19] M. L. Damiani, E. Bertino, B. Catania, and P. Perlasca. Geo-rbac: A spatially aware rbac. *ACM Trans. Inf. Syst. Secur.*, 10(1), Feb. 2007.

[20] Erin McCann. EHR vendor to report HIPAA breach. *Government Health IT*, Mar 2013.

[21] Federal Deposit Insurance Corporation. Privacy Act Issues under Gramm-Leach-Bliley, 1999. `http://www.fdic.gov/regulations/laws/rules/8000-4000.html`.

[22] D. F. Ferraiolo and D. R. Kuhn. Role-Based Access Control. In *15th National Computer Security Conference*, 1992.

[23] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli. Proposed NIST Standard for Role-Based Sccess Control. *ACM Trans. Inf. Syst. Secur.*, 4(3):224–274, Aug. 2001.

[24] M. Fowler. *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3 edition, 2003.

[25] Google. Google Health, 2011. `https://www.google.com/health`.

[26] V. C. Hu, D. Ferraiolo, and D. R. Kuhn. Assessment of Access Control Systems, September 2006. NIST Interagency Report 7316, `http://csrc.nist.gov/publications/nistir/7316/NISTIR-7316.pdf`.

[27] V. C. Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone. Guide to Attribute Based Access Control (ABAC) Definition and Considerations, January 2014. `http://nvlpubs.nist.gov/nistpubs/specialpublications/NIST.sp.800-162.pdf`.

[28] V. C. Hu and K. Scarfone. Guidelines for Access Control System Evaluation Metrics, September 2012. NIST Interagency Report 7874, `http://csrc.nist.gov/publications/nistir/ir7874/nistir7874.pdf`.

[29] J. Huang, D. M. Nicol, R. Bobba, and J. H. Huh. A framework integrating attribute-based policies into role-based access control. In *Proceedings of the 17th ACM symposium on Access Control Models and Technologies*, SACMAT '12, pages 187–196, New York, NY, 2012. ACM.

[30] J. Ingalsbe, L. Kunimatsu, T. Baeten, and N. Mead. Threat modeling: Diving into the deep end. *Software, IEEE*, 25(1):28–34, 2008.

[31] X. Jin, R. Sandhu, and R. Krishnan. Rabac: role-centric attribute-based access control. In *Proceedings of the 6th international conference on Mathematical Methods, Models and Architectures for Computer Network Security: computer network security*, MMM-ACNS'12, pages 84–96, Berlin, Heidelberg, 2012. Springer-Verlag.

[32] B. W. John Bethencourt, Amit Sahai. Advanced crypto software collection, Feb 2011. `http://acsc.cs.utexas.edu/cpabe/`.

[33] J. Joshi, E. Bertino, U. Latif, and A. Ghafoor. A generalized temporal role-based access control model. *Knowledge and Data Engineering, IEEE Transactions on*, 17(1):4 – 23, Jan 2005.

[34] C. Knouse. HP Praesidium Authorization Server, June 1998. `http://www.opengroup.org/security/meetings/sep97/pas9-97.pdf`.

[35] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of computation*, 48(177):203–209, 1987.

[36] N. Koblitz, A. Menezes, and S. Vanstone. The state of elliptic curve cryptography. *Designs, Codes and Cryptography*, 19(2-3):173–193, 2000.

[37] D. R. Kuhn, E. J. Coyne, and T. R. Weil. Adding attributes to role-based access control. *Computer*, 43(6):79–81, June 2010.

[38] B. Lang, I. Foster, F. Siebenlist, R. Ananthakrishnan, and T. Freeman. A flexible attribute based access control method for grid computing. *Journal of Grid Computing*, 7:169–180, 2009.

[39] H. W. Lenstra Jr. Factoring integers with elliptic curves. *Annals of mathematics*, pages 649–673, 1987.

[40] F. Li, D. Weerasinghe, D. Patel, and M. Rajarajan. An user-centric attribute based access control model for ubuquitous environments. In J. Zhang,

J. Wilkiewicz, and A. Nahapetian, editors, *Mobile Computing, Applications, and Services*, volume 95 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 361–367. Springer Berlin Heidelberg, 2012.

[41] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou. Fuzzy keyword search over encrypted data in cloud computing. In *IEEE Conference on Information Communications (INFOCOM'10)*, San Diego, 2010.

[42] B. Lynn. *On the Implementation of Pairing-Based Cryptosystems*. PhD thesis, Stanford University, 2007.

[43] B. Lynn. The pairing-based cryptography library, Feb 2011. `http://crypto.stanford.edu/pbc/`.

[44] Marc Winger. HIPAA Increases Financial Penalties For Repeat Violations To Address Increasing Healthcare Data Breaches. *Zephyr Networks*, Feb 2013.

[45] Marco Morana and Tony UcedaVelez. Threat modeling of banking malware-based attacks using the P.A.S.T.A. framework, 2011. `https://www.owasp.org/index.php/AppSecEU2011`.

[46] R. Martin. Group Selection and Key Management Strategies for Ciphertext-Policy Attribute-Based Encryption. Master's thesis, Rochester Institute of Technology, 2013.

[47] A. Masoumzadeh and J. B. Joshi. Purbac: Purpose-aware role-based access control. In *Proceedings of the OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008. Part II on On the Move to Meaningful Internet Systems*, OTM '08, pages 1104–1121, Berlin, Heidelberg, 2008. Springer-Verlag.

[48] V. Menzies-Flint. *Oracle Fusion Middleware Configuration Guide for Oracle Enterprise Repository*, chapter Configuring Advanced Role-based Access Control. Oracle, August 2012. `http://docs.oracle.com/cd/E14571_01/doc.1111/e16580/toc.htm`.

[49] Microsoft. Microsoft HealthVault, 2011. `http://www.healthvault.com/personal/index.aspx`.

[50] Microsoft Corporation. Threat Modeling, 2013. `http://msdn.microsoft.com/en-us/library/ff648644.aspx`.

[51] Microsoft Corporation. Threat Modeling Web Applications, 2013. http://msdn.microsoft.com/en-us/library/ff648006.aspx.

[52] Microsoft Corporation. Threats and Countermeasures, 2013. http://msdn.microsoft.com/en-us/library/ff648641.aspx.

[53] Microsoft Corporation. Web Application Security Frame, 2013. http://msdn.microsoft.com/en-us/library/ff649461.aspx.

[54] V. S. Miller. Use of elliptic curves in cryptography. In *Advances in CryptologyCRYPTO85 Proceedings*, pages 417–426. Springer, 1986.

[55] M. Nabeel, N. Shang, J. Zage, and E. Bertino. Mask: a system for privacy-preserving policy-based access to published content. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, SIGMOD '10, pages 1239–1242, New York, NY, 2010. ACM.

[56] S. Narayan, M. Gagné, and R. Safavi-Naini. Privacy preserving EHR system using attribute-based infrastructure. In *Proc. 2010 ACM Cloud Computing Security Workshop*, 2010.

[57] National Alliance for Health Information Technology. Defining key health information technology terms, 2008. http://healthit.hhs.gov.

[58] C. News. Doctor probed for improper health record access, Dec 2011.

[59] Q. Ni, A. Trombetta, E. Bertino, and J. Lobo. Privacy-aware role based access control. In *Proceedings of the 12th ACM symposium on Access control models and technologies*, SACMAT '07, pages 41–50, New York, NY, 2007. ACM.

[60] Octotrike. Trike Threat Model, 2013.

[61] S. Oh and S. Park. Task-role-based access control model. *Information Systems*, 28(6):533 – 562, 2003.

[62] S. Osborn, R. Sandhu, and Q. Munawer. Configuring role-based access control to enforce mandatory and discretionary access control policies. *ACM Trans. Inf. Syst. Secur.*, 3(2):85–106, May 2000.

[63] J. S. Park and R. Sandhu. Smart certificates: Extending x.509 for secure attribute services on the web. In *In Proceedings Of 22nd National Information Systems Security Conference (Nissc)*, pages 18–21, 1999.

[64] Practice Fusion. Web-based electronic health records, 2011. http://www.practicefusion.com.

[65] Privacy Rights Clearinghouse. Chronology of Data Breaches: Security Breaches 2005—Present, May 2014. Accessed May 4, 2014., http://www.privacyrights.org/data-breach.

[66] PwC's Health Research Institute. Old data learns new tricks: Managing patient security and privacy on a new data-sharing playground, September 2011.

[67] QNext Blog. Data Breach Stats: Medical/Healthcare, November 2013. http://qnext.com/blog/data-breach-stats-medicalhealthcare.

[68] I. Ray and M. Toahchoodee. A spatio-temporal role-based access control model. In *Proceedings of the 21st annual IFIP WG 11.3 working conference on Data and applications security*, pages 211–226, Berlin, Heidelberg, 2007. Springer-Verlag.

[69] Rochester General Health System. Rochester General Hospital General Statistics, May 2014. http://www.rochestergeneral.org/healthcare-professionals/physician-services/about-rochester/how-measure-up/.

[70] K. Roney. Titus Regional Medical Center Nurse Fired Over HIPAA Violation. *Beckers Hospital Review*, Jan 2012.

[71] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Proc. Advances in Cryptology - EUROCRYPT 2005*. Springer, 2005.

[72] L. Sainan. Task-role-based access control model and its implementation. In *Education Technology and Computer (ICETC), 2010 2nd International Conference on*, volume 3, pages V3–293 –V3–296, June 2010.

[73] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman. Role-based access control models. *Computer*, 29(2), Feb 1996.

[74] R. Sandhu, D. Ferraiolo, and R. Kuhn. The nist model for role-based access control: towards a unified standard. In *Proceedings of the fifth ACM workshop on Role-based access control*, RBAC '00, pages 47–63, New York, NY, 2000. ACM.

[75] R. Sandhu and Q. Munawer. How to do discretionary access control using roles. In *Proceedings of the third ACM workshop on Role-based access control*, RBAC '98, pages 47–54, New York, NY, 1998. ACM.

[76] B. Schneier. Attack Trees. *Dr. Dobb's Journal of Software Tools*, 24(12), Dec 1999.

[77] A. Shamir. Identity-based cryptosystems and signature schemes. In *Proc. CRYPTO 84 on Advances in Cryptology*. Springer, 1984.

[78] N. Shang, M. Nabeel, F. Paci, and E. Bertino. A privacy-preserving approach to policy-based content dissemination. In *Data Engineering (ICDE), 2010 IEEE 26th International Conference on*, pages 944 –955, March 2010.

[79] H. Shen and F. Hong. An attribute-based access control model for web services. In *Parallel and Distributed Computing, Applications and Technologies, 2006. PDCAT '06. Seventh International Conference on*, pages 74 –79, Dec 2006.

[80] V. Shmatikov and M.-H. Wang. Measuring relationship anonymity in mix networks. In *Proceedings of the 5th ACM workshop on Privacy in electronic society*, WPES '06, pages 59–62, New York, NY, 2006. ACM.

[81] S. Silow-Carroll, J. N. Edwards, and D. Rodin. Using electronic health records to improve quality and efficiency: the experiences of leading hospitals. *Issue Brief (Commonw Fund)*, 17:1–40, 2012.

[82] N. B. Sonia Jahid. PIRATTE: Proxy-based Immediate Revocation of ATTribute-based Encryption, April 2013. `https://bitbucket.org/hatswitch/piratte`.

[83] J. M. Spivey. *The Z Notation: A Reference Manual*. Prentice Hall International (UK) Ltd., Hertfordshire, UK, UK, 1992.

[84] B. Stepien, S. Matwin, and A. Felty. Advantages of a Non-Technical XACML Notation in Role-Based Models. In *9th Annual International Conference on Privacy, Security, and Trust*. IEEE, 2011.

[85] F. Swiderski and W. Snyder. *Threat Modeling*. Microsoft Press, Redmond, WA, USA, 2004.

[86] The Apache Software Foundation. Apache Cassandra, 2009. `http://cassandra.apache.org/`.

[87] The Apache Software Foundation. Apache Hadoop, 2014. http://hadoop.apache.org/.

[88] The Open Web Application Security Project (OWASP). Application Threat Modeling, 2013.

[89] The Organization for the Advancement of Structured Information Standards (OASIS). OASIS eXtensible Access Control Markup Language (XACML), 2013.

[90] M. V. Tripunitara and N. Li. A theory for comparing the expressive power of access control models. *J. Comput. Secur.*, 15(2):231–272, Apr. 2007.

[91] United States Department of Health & Human Services. The HIPAA Privacy Rule, 2002. http://www.hhs.gov/ocr/privacy/hipaa/administrative.

[92] United States Department of Health & Human Services. The HIPAA Security Rule, 2003. http://www.hhs.gov/ocr/privacy/hipaa/administrative.

[93] United States Department of Health & Human Services - HITECH. HITECH Act Enforcement Interim Final Rule, 2011. http://www.hhs.gov.

[94] United States Department of Health & Human Services - Privacy Rule. Standards for Privacy of Individually Identifiable Health Information; Final Rule, 2002. http://www.hhs.gov/ocr/privacy/hipaa/administrative/privacyrule/privrulepd.pdf.

[95] US Government Printing Office. Code of Federal Regulations. Title 45 - Part 160 - General Administrative Requirements, Subpart A, Sec. 160.103, 2007. http://www.gpo.gov/fdsys/pkg/CFR-2007-title45-vol1/xml/CFR-2007-title45-vol1.xml.

[96] US Government Printing Office. Code of Federal Regulations. Title 45 - Part 164 - Security and Privacy, Subpart C, Sec. 164.304, 2007. http://www.gpo.gov/fdsys/pkg/CFR-2007-title45-vol1/xml/CFR-2007-title45-vol1.xml.

[97] H. van der Linden, D. Kalra, A. Hasman, and J. Talmon. Inter-organizational future proof ehr systems: A review of the security and privacy related issues. *International Journal of Medical Informatics*, 78(3):141 – 160, 2009.

[98] J. Vijayan. Three fired for accessing records of tucson shooting victims. *Computerworld*, Jan 2011.

[99] XACML-V3.0. eXtensible Access Control Markup Language (XACML) Version 3.0. *OASIS Standard*, 2014.

[100] Z. Xu and S. D. Stoller. Algorithms for mining meaningful roles. In *Proceedings of the 17th ACM symposium on Access Control Models and Technologies*, SACMAT '12, pages 57–66, New York, NY, 2012. ACM.

[101] E. Yuan and J. Tong. Attributed Based Access Control (ABAC) for Web Services. In *Proceedings of the IEEE International Conference on Web Services*, ICWS '05, pages 561–569, Washington, DC, 2005. IEEE Computer Society.

[102] J. Zhu and W. Smari. Attribute based access control and security for collaboration environments. In *Aerospace and Electronics Conference, 2008. NAECON 2008. IEEE National*, pages 31 –35, July 2008.